Data-driven Natural Language Generation Using Statistical Machine Translation and Discriminative Learning

Elena Manishina

Laboratoire d'Informatique d'Avignon
Centre d'Enseignement et de Recherche en Informatique
Université d'Avignon et des Pays de Vaucluse

This dissertation is submitted for the degree of

Doctor of Philosophy

LIA/CERI/UAPV December 2015

Soutenue le 05 février 2016

Abstract

The humanity has long been passionate about creating intellectual machines that can freely communicate with us in our language. Most modern systems communicating directly with the user share one common feature: they have a dialog system (DS) at their base. As of today almost all DS components embraced statistical methods and widely use them as their core models. Until recently Natural Language Generation (NLG) component of a dialog system used primarily hand-coded generation templates, which represented model sentences in a natural language mapped to a particular semantic content. Today data-driven models are making their way into the NLG domain. In this thesis, we follow along this new line of research and present several novel data-driven approaches to natural language generation. In our work we focus on two important aspects of NLG systems development: building an efficient generator and diversifying its output. Two key ideas that we defend here are the following: first, the task of NLG can be regarded as the translation between a natural language and a formal meaning representation, and therefore, can be performed using statistical machine translation techniques, and second, corpus extension and diversification which traditionally involved manual paraphrasing and rule crafting can be performed automatically using well-known and widely used synonym and paraphrase extraction methods. Concerning our first idea, we investigate the possibility of using NGRAM translation framework and explore the potential of discriminative learning, notably Conditional Random Fields (CRF) models, as applied to NLG; we build a generation pipeline which allows for inclusion and combination of different generation models (NGRAM and CRF) and which uses an efficient decoding framework (finite-state transducers' best path search). Regarding the second objective, namely corpus extension, we propose to enlarge the system's vocabulary and the set of available syntactic structures via integrating automatically obtained synonyms and paraphrases into the training corpus. To our knowledge, there have been no attempts to increase the size of the system vocabulary by incorporating synonyms. To date most studies on corpus extension focused on paraphrasing and resorted to crowd-sourcing in order to obtain paraphrases, which then required additional manual validation often performed by system developers. We prove that automatic corpus extension by means of paraphrase extraction and validation is just as effective as crowd-sourcing, being at the same time less costly in terms of development time and resources. During intermediate experiments our generation models showed a significantly better performance than the phrase-based baseline model and appeared to be more robust in handling unknown combinations of concepts than the current in-house rule-based generator. The final human evaluation confirmed that our data-driven NLG models is a viable alternative to rule-based generators.

Abstract

L'humanité a longtemps été passionnée par la création de machines intellectuelles qui peuvent librement intéragir avec nous dans notre langue. Tous les systèmes modernes qui communiquent directement avec l'utilisateur partagent une caractéristique commune: ils ont un système de dialogue à la base. Aujourd'hui pratiquement tous les composants d'un système de dialogue ont adopté des méthodes statistiques et les utilisent largement comme leurs modèles de base. Jusqu'à récemment la génération de langage naturel (GLN) utilisait pour la plupart des patrons/modèles codés manuellement, qui représentaient des phrases types conçues pour des réalisations sémantiques particulières. C'était le cas jusqu'à ce que les approches statistiques se sont repandues dans la communauté de recherche en systèmes de dialogue. Dans cette thèse, nous suivons cette ligne de recherche et présentons une nouvelle approche à la génération de la langue naturelle. Au cours de notre travail, nous nous concentrons sur deux aspects importants du développement des systèmes de génération: construire un générateur performant et diversifier sa production. Deux idées principales que nous défendons ici sont les suivantes: d'abord, la tâche de GLN peut être vue comme la traduction entre une langue naturelle et une représentation formelle de sens, et en second lieu, l'extension du corpus qui impliquait traditionnellement des paraphrases définies manuellement et des règles spécialisées peut être effectuée automatiquement en utilisant des méthodes automatiques d'extraction des synonymes et des paraphrases bien connues et largement utilisées. En ce qui concerne notre première idée, nous étudions la possibilité d'utiliser le cadre de la traduction automatique basé sur des modèles ngrams; nous explorons également le potentiel de l'apprentissage discriminant (notamment les champs aléatoires markoviens) appliqué à la GLN; nous construisons un système de génération qui permet l'inclusion et la combinaison des différents modèles et qui utilise un cadre de décodage efficace (automate à état fini). En ce qui concerne le second objectif, qui est l'extension du corpus, nous proposons d'élargir la taille du vocabulaire et le nombre de l'ensemble des structures syntaxiques disponibles via l'intégration des synonymes et des paraphrases. À notre connaissance, il n'y a pas eu de tentatives d'augmenter la taille du vocabulaire d'un système de GLN en incorporant les synonymes. À ce jour, la plupart des études sur l'extension du corpus visent les paraphrases et recourent au crowdsourcing pour les obtenir, ce qui nécessite une validation supplémentaire effectuée par les développeurs du système. Nous montrons que l'extension du corpus au moyen d'extraction automatique de paraphrases et la validation automatique sont tout aussi efficaces, étant en même temps moins coûteux en termes de temps de développement et de ressources. Au cours d'expériences intermédiaires nos modèles ont montré une meilleure performance que celle obtenue par le modèle de référence basé sur les syntagmes et se sont révélés plus robustes, pour le traitement des combinaisons inconnues de concepts, que le générateur à base de règles. L'évaluation humaine finale a prouvé que nos modèles de génération représentent une alternative solide au générateur à base de règles.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Dialog systems

The humanity has long been passionate about creating intelligent machines that can freely communicate with us in our language. Even a much less ambitious perspective of task-specific communication in a human language with electronic devices and services has been pretty exciting. And if the former still has a long way to go, the later is becoming less illusive with the advent and advancement of various natural language processing (NLP) technologies. Most of us used at least once in our lives (or will undoubtedly use in the future) some of the following NLP technologies: Google Services (Search, Translate, Voice, etc)[1], Apples's Siri[2], Microsoft's Cortana[3], OpenEars[4], etc...

The systems communicating directly with users in a natural language share one common feature: they represent a form of a dialog system. It can be more or less sophisticated; it can use different modes of communication (oral, written, graphic, etc.), but the pipeline is always the same: it takes an input (a command/query/question) from the user in a natural language, processes it and provides a response, likewise in a natural language.

Modern dialog systems have a fairly standard architecture, which comprises five main components (see Figure 1.1): first the audio signal issued by the user is processed by a speech recognition (SR) module which produces a transcription in the form of a string of words; the transcription is passed to a speech (spoken language) understanding module (SLU) which extracts the meaning of the user input by tagging the transcription with semantic concepts. The set of concepts must be well defined in advance and the SLU tagger is trained on and

[1]www.google.com

[2]www.apple.com/ios/siri/

[3]embedded in Windows 10

[4]www.politepix.com/openears/

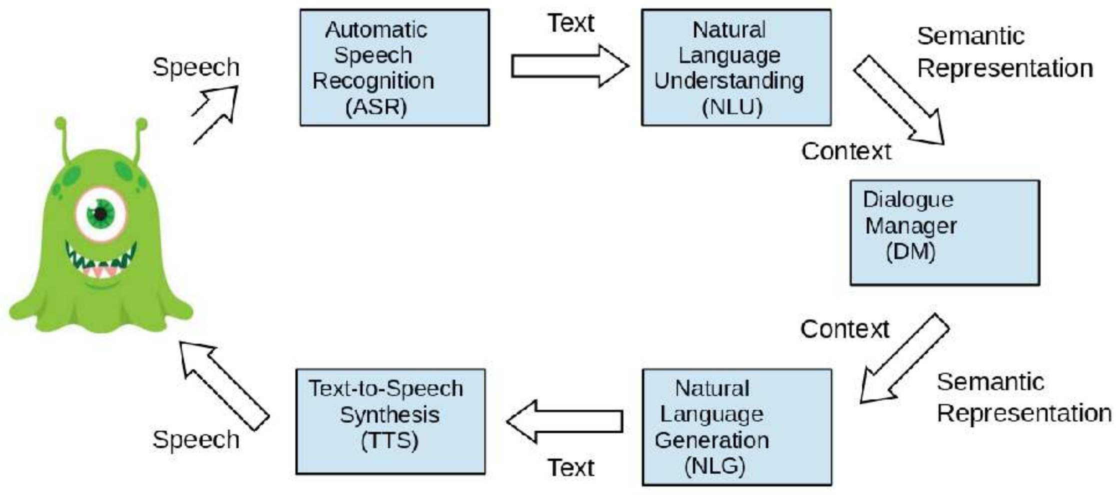

Speech
Automatic Speech Recognition (ASR)
Text
Natural Language Understanding (NLU)
Semantic Representation
Context
Dialogue Manager (DM)
Context
Semantic Representation
Natural Language Generation (NLG)
Text
Text-to-Speech Synthesis (TTS)
Speech

Fig. 1.1 Architecture of a Dialog System

adapted to specific data annotated with these concepts. The semantic representation of the input is passed to the dialog manager (DM) which determines the type of the utterance (request, negate, etc.), registers the units of information passed with the input and constructs a corresponding output, normally in the same form as the input, i. e. an ordered/unordered set of semantic concepts. Thus both the input and the output of the DM use the same semantic representation. After that the DM passes the output to a natural language generation (NLG) module which in turn generates a system reply in a human language from the set of semantic concepts supplied by the dialog manager. Finally the output of the NLG component (in a form of a natural language sentence) goes to a speech synthesizer (TTS) which converts it into an audio signal to be transmitted back to the user.

As of today almost all dialog system components embraced statistical methods and widely use them as their core models. The first to adopt a full-scale data-driven approach was the SR module: with the advent of Hidden Markov models (Baum and Petrie, 1966; Baum et al., 1970) in the late 60s the earlier rule-based SR systems, which used word templates and looked for sound patterns in the input, were able to pass from recognizing just about several hundred or thousand words to the ability of treating a potentially unlimited number of words.

In TTS both concatenative[5] and formant[6] synthesis models were followed by Hidden Markov Models (HMM)-based synthesis where the frequency spectrum, fundamental frequency, and duration of speech are modeled simultaneously by HMMs (Yoshimura et al.,

[5]Concatenative synthesis – TTS model which uses concatenated segments of recorded (human) speech.
[6]Formant synthesis – TTS model which couples additive synthesis and an acoustic model to produce vocal tracks of artificial speech.

1999). Speech waveforms are then generated from HMMs themselves based on the maximum likelihood estimation.

In SLU the Hidden Vector State (HVS) models (He and Young, 2003) and later CRF models (Hahn et al., 2008, 2011; Lefèvre, 2007) outperformed previous hand-coded methods not just in terms of accuracy, but also in terms of saving human efforts and resources (De Mori et al., 2008).

In dialog management hand-crafting complex decision rules was replaced by reinforcement learning[7] (Paek, 2006; Walker, 2000) and by acquiring decision-making strategies from observation and imitation of humans (Chandramohan et al., 2011; Passonneau et al., 2011).

In our work we focus on NLG, more specifically on **tactical** generation which handles the production of actual utterances in human language from a formal semantic representation, as opposed to **strategic** generation, which is the process of generating the meaning to be expressed in a chosen semantic formalism; in our setup this task is performed by the dialog manager.

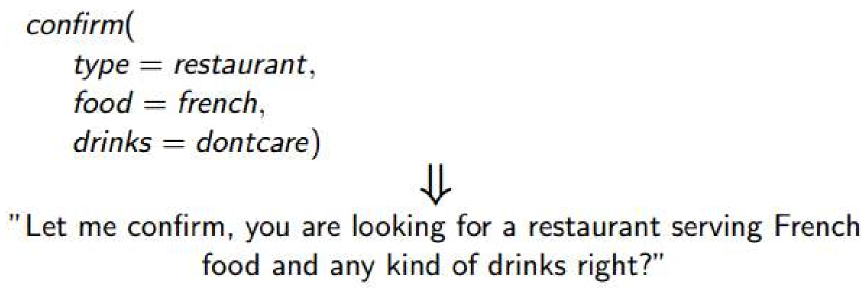
Until recently NLG module primarily used hand-coded templates, which represented model sentences, optionally containing slots for variable values, and which were mapped to a particular semantic realization (see sections 2.1.4 and 3.3). Templates are still widely used especially in written language generation and also for very domain-specific tasks which do not require variability and flexibility in the output. More dynamic spoken NLG modules required another approach, which would not just allow for more flexibility, but would also be less expensive in terms of development time and resources. In that respect data-driven paradigms for language generation looked particularly promising. In the following sections we take a closer look at the task of NLG and various generation paradigms.

1.2 Natural Language Generation (NLG)

Natural language generation is the area of natural language processing which handles the production of natural language utterances (either written or spoken) from a machine representation form: conceptual, logical, symbolic, etc. (see Fig. 1.2).

As we said earlier, the NLG module was one of the last DS components to discover the virtues of statistics. For quite a long time a common way to implement the module was to build generation templates, i.e. predefined model sentences in a human language which replaced specific statements in a formal semantic representation during generation. Templates are often conceived for a concrete task in a given domain and are well-suited for

[7]In reinforcement learning scenario the dialog is represented as a Markov decision process - a process where, in each state, the DM has to select an action, based on the state and the possible rewards from each action.



confirm(
    type = *restaurant*,
    food = *french*,
    drinks = *dontcare*)

$\Downarrow$

"Let me confirm, you are looking for a restaurant serving French food and any kind of drinks right?"

Fig. 1.2 NLG system input and output

complex semantic formalisms and/or narrow technical domains. Template-based models have a number of advantages: they are robust and reliable; the output produced by these modules is likely to be grammatically correct and not to contain unexpected generation errors (though there are exceptions to that claim, see section 2.1.2); the output is supposed to be easily predictable and the generation process fully controlled. Despite all the above, there are reasons to challenge template-based NLG modules' capacity, especially in spoken dialog systems. First of all, templates are constructed manually which means that their production requires considerable time and human resources; templates are not automatically extendable and not able to handle unknown inputs (e.g. unseen combinations of semantic concepts); they are not flexible - the same template is used in many different situations and in different contexts; templates often sound unnatural due to their generic structures and lack of expressiveness (which stems from their ability to be used in various contexts); and finally a template-based model in its pure form is not able to adapt to the user and is not able to learn. These features can indeed be added to a template-based model, but the time and effort spent on coding and further maintenance would not be negligible. Data-driven models do away (at least partially) with these issues; nevertheless they face many challenges of their own.

Notably, building a data-driven NLG model requires training data; thus the first step in developing such systems is building the corpus to learn from, which in turn comes to creating something similar to templates. That raises the question: are statistical models that effortless as their advocates claim? Though collecting the initial training data requires some effort and resources in the beginning, once the basic set of utterances is created it can be extended and refined automatically; moreover it can be adapted to a particular task or user, just like a general-domain machine translation system can be adapted to a specific domain (see (Koehn and Schroeder, 2007; Wang et al., 2012)). Extending the training corpus may consist in adding new training instances (modified user inputs, paraphrases), integrating synonyms into the system's lexicon, automatically searching for new generation patterns on the web, etc. Extending and diversifying the corpus that way improves the quality of interaction between the user and the system and enriches the conversation (the system's responses in the first place), at the same time sparing the developers coding time and effort. We take a closer look at corpus extension techniques in chapter 4.

1.2.1 What is a 'good' NLG system

The most important property of any generation module is the fluency of its output. This property sums up the whole purpose of an NLG system, which is to present the information to the user in an understandable and correct form.

What else can be expected from a 'good' NLG module, apart from output fluency? We can look at this problem from different perspectives:

- at the user level: among the most important system characteristics of a spoken NLG system are variety and diversity of produced phrases, as well as the ability to adapt to a particular user and situation (for example, taking into consideration gender, age and social status of the user if this information is available to the system);
- at the developer level: the system should be easy to implement, extend and maintain and also easy to transfer to a new task or corpus; the core NLG model should be extendable without the need for costly upgrade and ideally it should be generic enough and task-independent.

There exists another perspective which is of special interest to us - a scientific one. From the scientific perspective, we exploit the idea of language generation being the reverse of language understanding.

1.2.2 Language generation as the reverse of (spoken) language understanding (SLU)

The idea of NLG being the reverse of SLU and thus being able to use similar methods and models has recently been very popular within the dialog systems research community (see for example (Wong and Mooney, 2007)).

Indeed SLU and NLG look like being one and the same process but moving in opposite directions. In the present work we adopt the same position in general, but we believe there are some observations to be made:

- Data-driven SLU task is regarded by many researches as a tagging problem. Semantic formalisms used by an SLU module to produce the output normally have a fixed set of semantic units (concepts, frames, etc); SLU taggers based on algorithms like

Conditional Random Fields deal with such limited tasks quite well. Generation output is virtually limitless in its variety. So NLG can hardly be viewed as a sort of simple tagging. Rather it may be regarded as a translation task, the machine representation being the source and natural language utterances being the target sides for the translation system (see section 1.3.2).

- The unit order in the output of an SLU module is of no importance to the dialogue manager which receives this output, nor to the system developer wishing to examine it. The only concern during evaluation of an SLU module is the presence of all semantic units in the output. The output of an NLG module is intended for a final user and, consequently, its form is just as important as its content. In many cases additional processing and decoding steps (like reordering and the use of an n-gram language model) might be necessary in order to produce grammatically correct and fluent output.

Based on the above observations we can derive some key properties of an NLG model that we want to obtain:

1. The model should be generic and suitable for any language and task. Minor modifications and/or extensions in the semantic formalism or the training corpus should not affect the generation model itself; ideally all changes are to be learned and integrated automatically;

2. The NLG model should be able to process unseen inputs, in our case unseen set of semantic concepts. This property is one of the key differences between the template-based and statistical NLG models;

3. The system should be extendable. In case of data-driven models it often means additional manual annotation of the training examples and extra development efforts. We believe that corpus extension can be performed automatically. We elaborate on the idea of automatic corpus extension in chapter 4.

4. Ideally a unique training paradigm should be used in both SLU and NLG modules which are thought to be the same process moving in opposite directions.

Taking into consideration the above observations we opted for a data-driven approach to language generation, which suits the best our objectives.

1.3 Data-driven NLG

1.3.1 Data-driven NLG models

Currently data-driven models dominate the NLG research community; that is especially true in case of wider domain tasks where it is virtually impossible to hand-craft the templates for all possible combinations of semantic units and their respective surface realizations, not even mentioning the possibility of drafting several different surface realizations for each combination.

There have been several successful attempts to apply data-driven techniques to NLG. Here we will briefly outline some of them (for a more detailed account of these and other NLG models see chapter 2): BAGEL NLG system (Mairesse et al., 2010) uses Dynamic Bayesian Networks to generate output strings in a natural language from unordered stacks of semantic concepts; (Dinu and Baroni, 2014) suggests that the generation can be performed using distributional semantic models; (Swanson et al., 2014) performs generation via language modeling with specific vocabulary constraints; following a number of studies on re-ranking in NLG system's output, (Konstas and Lapata, 2012) further investigates the potential of discriminative re-ranking. It is worth noting that most studies lie within the time frame of the last 5 years which proves the grown interest in data-driven language generation in recent times.

1.3.2 NLG as a translation task

It makes sense to view the generation task as a kind of translation between the semantic/conceptual representation and the natural language utterances. (Wong, 2007; Wong and Mooney, 2007) suggest that both SLU and NLG tasks can be performed using state-of-the-art machine translation (MT) techniques, more specifically the synchronous parsing algorithm used in syntax-based MT frameworks. It is worth noting that this technique requires annotation of the training data at the syntactic level. (Langner et al., 2010) used the state-of-the-art phrase-based translation paradigm as a core generation model; he proved that even a standard (classical) MT pipeline can be suitable for NLG and shows a decent performance comparable to that of rule-base generators, using at the same time all the benefits of an SMT system, like, for example, the ability to treat unseen combinations of concepts.

In the present work we continue along this line of research, i.e. we intend to cast the problem of language generation as a translation task. Also in order to create a unique SLU/NLG paradigm we investigate the possibility of building a discriminative NLG model based on Conditional Random Fields (CRF) and integrating this model into a larger translation framework.

1.4 Thesis contribution

As we said earlier, data-driven NLG raises several challenges, among which are: building the training corpus, ensuring a tighter control over generation process, automation of model/corpus extension and possibly adding user adaptation capabilities. In out work we address some of these issues; we propose several novel approaches to language generation and explore the possibilities of automatic corpus extension.

1.4.1 Our generation task

Concerning the generation models, we investigate the idea of using n-gram translation framework and explore the potential of discriminative learning (notably Conditional Random Fields) as applied to NLG; we build a generation pipeline which allows for inclusion and combination of different models, notably n-gram and CRF couple, and which uses an efficient decoding procedure.

Regarding corpus extension, to our knowledge, there have been no attempts to automatically augment the size of the system vocabulary via integrating synonyms; most studies on corpus extension have resorted to crowd-sourcing in order to obtain paraphrases, which then required manual validation, often performed by system developers. We prove that automatic corpus extension by means of incorporating synonyms into the system's vocabulary and automatic paraphrase extraction and integration are just as effective as crowd-sourcing, being at the same time less expensive.

Our generation pipeline has a number of advantages over previously developed NLG frameworks: it does not require additional corpus annotation or pre-processing (e.g syntactic parsing as in (Wong and Mooney, 2007)); it is language- and domain-independent; we propose a number of solutions for corpus extension which are fully automatic (unlike manual paraphrasing and validation in (Mairesse et al., 2010; Mitchell et al., 2014)). In the two following subsections we give a brief overview of the work presented in this thesis.

1.4.2 Core NLG models

Our objective is to find a universal approach to spoken language generation which could be applied to different generation scenarios without major modifications. For that purpose in this thesis we work with data sets in two different languages: English and French. Unlike many studies previously carried out in the NLG domain (e.g. (Wong and Mooney, 2007)) our study uses more complex data (see section 3.3) and our experimental setup involves an actual continuous interaction between a user and the system.

We believe that the generation process can be viewed as translation between the semantic representation of the intended meaning and actual words. We aim at using techniques developed in statistical machine translation (SMT) domain to perform language generation. We can put to good use such properties of SMT as: the ability to store and produce multiple translations for a given input unit, which adds the desired variability to the generated output; decomposition of a sentence into phrases to be translated separately, which allows for translation of unseen unit combinations in the input; the use of a target LM, that ensures a fluent output; reordering which is particularly useful for unordered sets of input units and which fosters grammaticalness and overall well-formedness of output sentences.

In the course of our work we investigate the potential of several translation models (TMs) which represent different SMT paradigms. As our baseline we use a classical state-of-the-art phrase-based translation framework as implemented in the Moses toolkit (Koehn et al., 2007). Then we turn to the n-gram based models. N-gram translation paradigm allows us to model joint probabilities of semantic concepts and their correspondent lexical realizations; it explicitly models the context, i.e. previously generated sequences of concept/words couples; finally it is a computationally cheap, yet effective modeling solution. The n-gram framework, as implemented in NCODE toolkit (Crego et al., 2011), becomes the holding frame for our generation pipeline, as it allows for an easy integration and combination of different components as well as adjustment of various parameters. Notably we integrate a CRF-based generation model into the framework in a form of an additional translation score. We selected CRFs as they represent a very powerful modeling paradigm and offer more freedom in feature selection. Our framework uses finite state transducer (FST) best-path search decoding that allows for transducer combination, which in turn makes it possible to perform a joint decoding and make fully informed decisions based on multiple models' estimations at decoding time. We evaluate the performance of our NLG models at the development stage using the BLEU metric. To assess the extended versions of the best-performing models and also in order to compare the performance of our systems with that of the rule-based generator we carry out an extensive human evaluation which involves several different user groups (Chapter 5).

1.4.3 Corpus extension

In order to make our NLG systems more user-friendly and more flexible we would like to implement several adds-on to the core generation model. These include extending the training corpus and adding a pragmatic/modal component, which is intended to make the system sound more 'human'. We extend the corpus by adding new phrases, that are similar to the ones already used for training, to the target (lexical) side of the corpus. These new phrases are automatically constructed paraphrases of the existing ones.

We create these paraphrases using the following techniques:

1. replacing open class words (nouns, verbs, adjectives, adverbs) with their synonyms automatically acquired from an on-line ontology (Ontology-based method) or extracted from a parallel corpus (PC method)

2. replacing sub-phrases inside a phrase with its paraphrase(s) automatically extracted from the parallel corpus (PC method)

Both methods are fully automatic and require only an initial manual validation.

As a further refinement to our models, we envision integrating a modal component responsible for a pragmatic/emotional facet of generated phrases. A modal is incorporated into the semantic representation of each dialog act in a form of an additional concept which has no informative content and only stores pragmatic information about a given dialog act. This information allows us to choose a specific modal clause at decoding time. The set of eligible modals for each pragmatic frame is learned from a corpus, i.e. obtained automatically.

1.5 Thesis outline

We start with the outlook on the work that has been carried out previously in the fields of Natural Language Generation and Machine Translation (Chapter 2). The first part of this chapter gives an overview of the NLG history from the first instance-based models to the modern data-driven approaches (sections 2.1.4, 2.1.4, 2.1.5). The second part provides theoretical background for the techniques that we used, notably Conditional Random Fields (CRF) (section 2.2.5) and N-gram machine translation (section 2.2.3). Chapter 3 is devoted to our core generation frameworks: its main components, experimental setup and results. Specifically, in this chapter we present the baseline phrase-based generation model (sections 3.2.1 and 3.4.1), CRF-based models (3.4.3), N-gram models (sections 3.2.2 and 3.4.2) and finally the combination of CRF and N-gram models into a unique pipeline (sections 3.2.3 and 3.4.4). We also discuss in detail such important issues in language generation as reordering (section 3.2.2) and basic generation units (vocabulary) acquisition (section 3.2.2). In Chapter 4 we describe our methodology for automatic corpus extension (sections 4.3.1 and 4.3.2) and integration of a pragmatic/modal component (4.3.3). In Chapter 5 we present the protocol and the results of the final large-scale human evaluation. Finally we conclude our study with a discussion of results and an outline of some future research directions.

Chapter 2

Natural Language Generation, Background and State-of-the-art

In this chapter we will give a theoretical and historic overview of the methods that we used to build our generation models. In the first part of this chapter we discuss the approaches which were (and some still are) widely used within the Dialog Systems and NLG communities, as well as more recent developments in the language generation domain (section 2.1). We start with an introductory section on two major generation tasks: written and spoken NLG (section 2.1.2) and the discussion of various semantic formalisms that are (or were at some point) widely used in the domain (Section 2.1.3).Then we provide a brief outline of different NLG paradigms, including rule-based models, semi- and fully data-driven models (section 2.1.4); then in section 2.1.5 we take a closer look at more specific data-driven paradigms, like SMT-based models and chart generation. We also discuss a number of important issues related to language generation: corpora collection (section 2.1.8), learning templates/grammars from a corpus (section 2.1.6) and user adaptation (section 2.1.7). We end this part with an overview of some well-established NLG evaluation techniques in section 2.1.9.

In the second part of this chapter we give the theoretical foundations of the underlying model components. We start with an introduction to statistical machine translation (section 2.2.1) and its core components: alignment, reordering, translation models and target language models (2.2.2). We take a closer look at the translation model and various SMT paradigms in corresponding subsections of section 2.2.3: phrase-based, syntax-based, factored SMT which is a variation of phrase-based models and finally n-gram SMT framework. Section 2.2.4 we present the minimum error rate training algorithm, which is generally used for learning the optimal parameters in log-linear models. In section 2.2.4 we discuss discriminative learning (DL) in SMT, specifically the DL model that we use in our NLG system, namely Conditional Random Fields. Finally we end this part with the section on MT evaluation (section 2.2.6).

2.1 NLG: previous work

2.1.1 Introduction

NLG is the process of automatically generating natural language utterances in textual or audio form from formal semantic representation[1]. Semantic representation may take different forms: a set of semantic concepts, a semantic tree, a logical statement, a database entry, etc.; it can represent a fixed semantic expression or contain variables – slots for dynamically inserted values (for the overview of the most widely used semantic formalisms see section 2.1.3). The choice of a particular semantic representation depends on the task at hand and on the system design. The semantic formalism for a given system is defined in advance and constitutes a finite set of units and instructions.

Besides various semantic formalisms, NLG systems use different generation paradigms. Based on the underlying model, NLG systems can be divided into three major groups: rule-based, semi-data-driven and fully data-driven[2].

In rule-based models generation patterns are coded manually. Normally such models are developed for restricted domains; they use a limited vocabulary and a small set of predefined syntactic structures in their templates. These models are robust, but lack flexibility and are too costly to develop and maintain.

Semi-data-driven models introduce some kind of statistical learning into pre-processing or post-processing steps, but the generation model core remains rule-based.

In fully data-driven models generation patterns are learned from a training corpus in a supervised ((Angeli et al., 2010; Mairesse et al., 2010)) or unsupervised ((Duma and Klein, 2013)) manner. The only manual work involved is reduced to preparation of the training data: annotation, alignment, validation, etc. Recent studies revealed a growing interest in diminishing the cost of corpus acquisition by either automation of the process, e. g. searching for patterns in a general corpus (Tadeu et al., 2010), or crowd-sourcing, i.e. spreading the task among a huge number of untrained annotators (Mitchell et al., 2014) (see section 2.1.8). A variety of data-driven methods have been proposed (section 2.1.4) , from dynamic Bayesian networks (Mairesse et al., 2010) to distributional semantics (Dinu and Baroni, 2014) and

[1]A number of alternative definitions have been previously proposed, e.g., one of them is «natural language generation is the process of deliberately constructing a natural language text in order to meet specific communicative goals (McDonald, 1992)». We do not adopt this definition in our work, as it lacks the «automatic» part and «meet specific communicative goals» refers more to dialog management (or dialog planning) than language generation in our setup.

[2]By ’fully data-driven’ we mean that the model itself is learned from training instances and not hard-coded in rules; corpus creation and evaluation might be, and in most cases are, manual.

chart generation (Wong and Mooney, 2007). Most of them have shown similar performance as rule-based models, at the same time offering more flexibility and variability in their output.

Another important distinction within the domain of NLG is the output mode: spoken VS written language. The distinction between the two is not always well-drawn when it comes to discussing the previous work in the field of NLG. And nevertheless these two domains have been developing in parallel, using different methods, different models and different semantic formalisms; also they pursue different goals and emphasize different key properties of their output: written NLG aims at precision and clarity in their output, while in spoken NLG the major concerns are comprehensiveness, understandability and fluency. What have little importance in written NLG - fluent language, diversity of the output, forms of politeness (that we refer to as a modality component in the present work) - is vital in spoken NLG. In the following sections (specifically 2.1.2 and 2.1.3), we aim at drawing a clear distinction between the two areas.

2.1.2 NLG tasks: written VS spoken language generation

As computer technology reaches out to more and more users in various domains, so grows the number of practical applications for NLG. This is partly because it offers an alternative way to access information, which turns out to be appealing to the end user as it uses a habitual channel of communication – natural language. At present, the area of NLG (particularly spoken language generation) is in active development and the number of applications using NLG in some form is expected to grow.

As we said earlier, there are two major forms (channels) of communication that an NLG system may use: written and spoken language. **Written NLG** handles text/document creation from structured semantic representation. It often represents a standalone application, i. e. it is not a part of a larger system (e. g. a dialog system), and thus is easier to maintain and upgrade without affecting the overall performance, e.g. by bringing modifications into semantic formalism. We can distinguish several types of written NLG systems[3]:

1. Automated document production. It consists in transforming a symbolic information representation into a human-readable text. Normally this text is well structured and comprehensive. A typical example of automated document production is FoG devel-

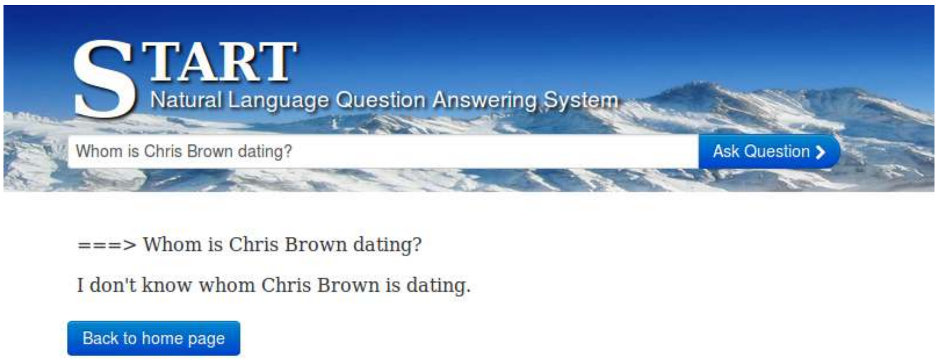
[3]Some researches refer to written NLG as a simple "report generation" or "document generation", which do not represent a real NLG as it does not go beyond a simplistic template-based text generation. We beg to differ (in line with the majority of researches in the area); we believe that any task that handles text production and involves at least some kind of linguistic processing at any level (syntactic, lexical, morphological) should be considered an NLG application.

oped by GoGenTex in 1992 ((Wanner, 2010)). It creates textual weather reports from numeric images and graphs (see section 2.1.3).

2. Presentation/summarization of information in a specific readable format. It is somewhat similar to automated document production; it 'translates' the semantics in the form of symbolic or logical statements into a textual form. Often it does not go beyond isolated statements (or a group of statements). The more complex and varied semantic formalism is, the more linguistic processing would be involved, and the closer it gets to a genuine NL generation. For example, TEMSIS by DMKI GmbH[4] generates the summaries of pollution reports from atomic data key-value pairs, like the following:

   (year 1986), (country Germany),(temperature((minimum((unit degrees-c) (number 05))) => In Germany in 1986 the minimum temperature was 5 C.

3. Question answering (QA) systems. QA systems provide natural language responses to user queries issued in a natural language. They might seem somewhat similar to dialog systems but, in fact, they do not involve continuous interaction between a user and the system; as a rule, there is no recording of previous states (no dialog history), each query is unique and is processed in isolation from previous ones. An example of such a system is START QA system (Katz et al., 2003).


START
Natural Language Question Answering System
Whom is Chris Brown dating?
Ask Question
Back to home page

===> Whom is Chris Brown dating?

I don't know whom Chris Brown is dating.

Fig. 2.1 START interface

The natural language processing component of START consists of two modules that share the same grammar. The understanding module analyzes the input text and produces a formal representation of the query that encodes the information found in it.

[4]www.lt-innovate.eu/directory/organisation/temis-gmbh

Given the query, the system searches the web for the answer and when it is found (or not found) the generation module produces the output sentences using the obtained information or paraphrasing the initial question, if the search result is *null*.

Though written NLG might look like an easy task, especially considering simple template text generation, it does have to deal with eventual disfluencies like the one in Fig.2.2 and thus written NLG systems often incorporate complex grammar rules and spell-checkers.

*There **have** been **1** weeks of rain caused by cyclone in the North-West.*

Fig. 2.2 Errors in the system's output (from 'Daily weather reports')

In spoken NLG the generated output is transferred to a speech synthesizer which converts it into an audio signal. It can be a one-time local generation or a continuous conversation within a dialog system.

1. Application-specific local (out-of-context) generation.
   In this scenario the generation system produces one-time utterances informing the user about something. The goal is to transmit the information through the medium of sound and each utterance is generated as a result of a user action or some specific conditions. Such systems are widely used in automated teaching and instruction generators, like **AutoTutor** (Graesser et al., 2005) system and **Atlas** (Freedman, 1999).

2. NLG within a dialog system.
   Spoken NLG module is often a part of a larger dialog system, which comprises several components. Thus all of them use the same semantic formalism and sometimes the same processing paradigm (like in case of SLU and NLG using the same underlying model). It implies that the NLG module needs to be adjusted to and function in accordance with other system components. Dialog systems often cover larger domains and allow for more variation in the NLG output, which makes NLG component vulnerable to grammatical errors and subsequently makes 'fluency' a particular concern (especially in data-driven NLG models).
   Such systems are widely used today in the commercial sectors. A typical example of a full-scale dialog system is **NICE** (Boye et al., 2004).

2.1.3 Semantic formalisms (SFs)

Semantics is the most important aspect of inter-human or human-machine interaction, natural language being merely an instrument of encoding the meaning we wish to transmit. An entire area of computational linguistics, called **computational semantics**, is concerned with the automated processing of meaning associated with natural language expressions. A huge number of different semantic formalisms have been developed. Still when it comes to a deployment of an NLG system, the specificity of the task often requires the development of a specific formalism, which may be based on an existing one or be completely new. A choice for a particular semantic formalism is of great importance; SFs determine the format of NLG models' input and as such can directly affect the quality of the output. A well thought out semantic specification can save developers a lot of time, eliminate ambiguities in the output, facilitate corpus collection and annotation. There exist various formats of NLG input depending on the task and also on the expected output. Particularly, the characteristics of the semantic representations differ significantly in spoken and written generation.

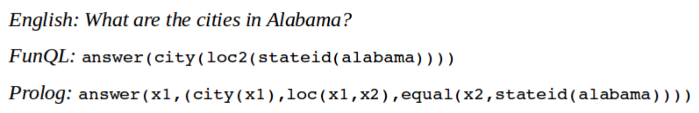
In written NLG an important characteristic of SF is precision, as well as how much information it contains and the quality of this information; the complexity it its form is rarely a concern; oftentimes, it is barely interpretable by humans (see Fig. 2.3). The output of a written NLG system is often evaluated against a reference text, which makes it unnecessary for humans (apart from system developers) to have a thorough understanding of the semantics. In spoken NLG (specifically data-driven), along with precision, an important characteristic of SFs is clarity and intuitiveness (i.e. how easy it is to obtain corpus annotations (in case of manual annotation) and how intuitive a given semantic formalism is for human annotators). The performance of spoken NLG systems is often assessed by human evaluators which makes the need for an intuitive SF even more apparent. In the rest of this section we outline some of the most common semantic formalisms for written and spoken NLG.

```
RUNID fiberall FIBER 6/19/93 act yes
FA 1301 2 1995
FA 1201 2 1995
FA 1401 2 1995
FA 1501 2 1995
ANF co 1103 2 1995 48
ANF 1201 1301 2 1995 24
ANF 1201 1301 2 1995 24
END. 856.0 670.2
```

Fig. 2.3 PlanDoc input (example from Claire Gardent). PlanDoc generates natural language reports from the interaction between telephone planning engineers.

1. **Logic forms (predicate structures)**. Logic forms are first-order logic knowledge representations of natural language sentences formed by the conjunction of concept predicates related through shared arguments. This formalism is one of the most

commonly used as well as the most intuitive (for humans) way to represent the meaning. A typical example of such formalism, FunQL, has been devised for the GEOQUERY domain (Kate et al., 2005).



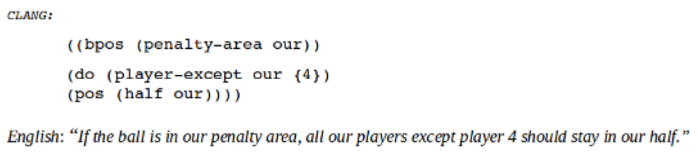
English: What are the cities in Alabama?

FunQL: `answer(city(loc2(stateid(alabama))))`

Prolog: `answer(x1,(city(x1),loc(x1,x2),equal(x2,stateid(alabama))))`

Fig. 2.4 FunQL example and the corresponding Prolog statement

Another example of logical semantic formalism is CLANG: the RoboCup Coach Language. RoboCup[5] is an international competition in robotic soccer. In this competition the coaches transmit the commands in CLANG to the robots-players. The formal description of CLANG can be found in (Chen et al., 2003).



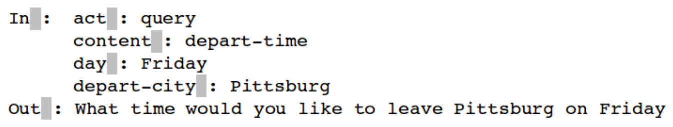
CLANG:

```
((bpos (penalty-area our))
(do (player-except our {4})
(pos (half our))))
```

English: "If the ball is in our penalty area, all our players except player 4 should stay in our half."

Fig. 2.5 Example in CLANG (from (Kate et al., 2005))

2. **Symbolic representations**. Formalisms of this group might encode various types of data: database entries, key-value pairs, numeric data, etc. It does not have any generic structure unique for all formalisms of this group. It bears no relation to a generated text whatsoever and it is hardly intelligible to humans (non experts). One example of such a formalism is the one used by CMU Communicator travel planning system (Oh and Rudnicky, 2000); it takes the input in a form of a frame of attribute-value pairs (Bennett and Rudnicky, 2002).



```
In :  act : query
      content : depart-time
      day : Friday
      depart-city : Pittsburg
Out : What time would you like to leave Pittsburg on Friday
```

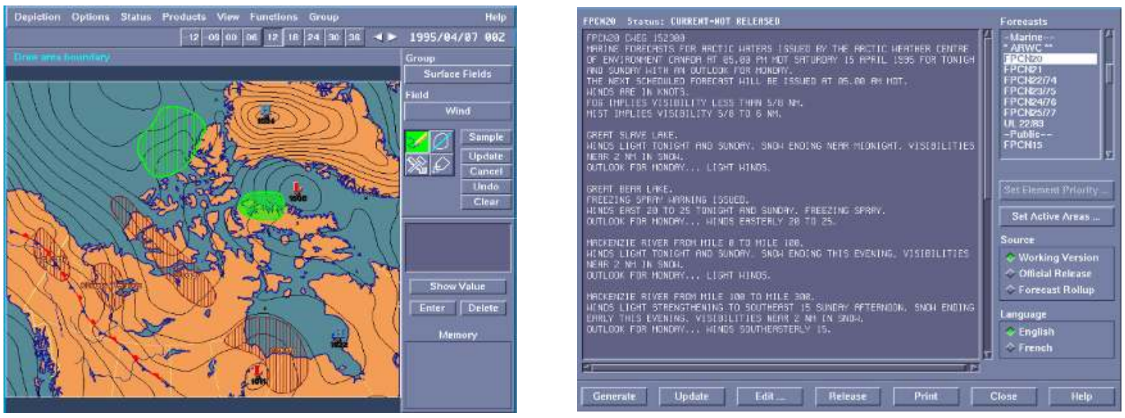
Fig. 2.6 CMU system input and output example

Another example is the input into the PLANDoc system (Shaw, 1995) which generates natural language reports based on the interaction between telephone planning engineers (Fig. 2.3).

[5]www.robocup.org

3. **Graphical representation (images)**. This formalism is typical of written NLG, specifically automated document production. It allows to quickly register complex phenomena, that are difficult to encode into a particular symbolic form, and pass it to an NLG system which in turn derives the necessary information and converts it into a human readable text. An example of such graphical formalism is the input into CoGenTex' s FoG[6] system which produces textual weather reports (Fig. 2.7).

(a) (b)



```
FPCN20  Status: CURRENT-NOT RELEASED
FPCN20 CWEG 152300
MARINE FORECASTS FOR ARCTIC WATERS ISSUED BY THE ARCTIC WEATHER CENTRE
OF ENVIRONMENT CANADA AT 05.00 PM MDT SATURDAY 15 APRIL 1995 FOR TONIGH
AND SUNDAY WITH AN OUTLOOK FOR MONDAY.
THE NEXT SCHEDULED FORECAST WILL BE ISSUED AT 05.00 AM MDT.
WINDS ARE IN KNOTS.
FOG IMPLIES VISIBILITY LESS THAN 5/8 NM.
MIST IMPLIES VISIBILITY 5/8 TO 6 NM.

GREAT SLAVE LAKE.
WINDS LIGHT TONIGHT AND SUNDAY. SNOW ENDING NEAR MIDNIGHT. VISIBILITIES
NEAR 2 NM IN SNOW.
OUTLOOK FOR MONDAY... LIGHT WINDS.

GREAT BEAR LAKE.
FREEZING SPRAY WARNING ISSUED.
WINDS EAST 20 TO 25 TONIGHT AND SUNDAY. FREEZING SPRAY.
OUTLOOK FOR MONDAY... WINDS EASTERLY 20 TO 25.

MACKENZIE RIVER FROM MILE 0 TO MILE 100.
WINDS LIGHT TONIGHT AND SUNDAY. SNOW ENDING THIS EVENING. VISIBILITIES
NEAR 2 NM IN SNOW.
OUTLOOK FOR MONDAY... LIGHT WINDS.

MACKENZIE RIVER FROM MILE 100 TO MILE 300.
WINDS LIGHT STRENGTHENING TO SOUTHEAST 15 SUNDAY AFTERNOON. SNOW ENDING
EARLY THIS EVENING. VISIBILITIES NEAR 2 NM IN SNOW.
OUTLOOK FOR MONDAY... WINDS SOUTHEASTERLY 15.
```

```
-Marine--
*ARWC**
FPCN20
FPCN21
FPCN22/74
FPCN23/75
FPCN24/76
FPCN25/77
UL 22/83
--Public--
FPCN15
```

Fig. 2.7 FoG input (a) and output (b) (pics. from Claire Gardent)

4. **Semantic trees/flattened trees**. This formalism is similar to a symbolic one, but it also uses more complex hierarchical relations (parent/child concepts). An example of such formalism is the one used in WeatherReporter (Reiter and Dale) (Fig.2.8).

```
((type daily weatherrecord)
              (date(
                      (day24) (month02)
       (year1992)
                      )
              )(temperature(
                      (minimum( (unit degrees-c)
       (number05) ) )
                      (maximum( (unit degrees-c)
       (number11) )
              )
       ) (rainfall( (unitmillimetres)
       (number3) ) )
       )
```

```
February the 24th, 1992 was cooler and
drier than average. The minimum
teperature was 5 C. The maximum
temperature was 11 C. The total rain
for the day was 3 mm. The total rain
for the year so far is well below
average.
```

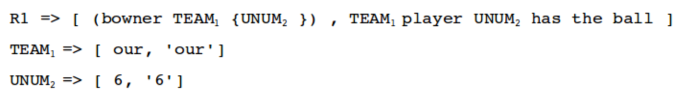
Fig. 2.8 Daily weather records input and output

5. **Context Free Grammar**. One of the most generic semantic formalisms widely used in different NLP applications, including NLG, is a Context Free Grammar (CFG). A CFG represents a quintuple, like in (2.1):

$$G = \{N, Te, Tf, L, S\} \tag{2.1}$$

[6] www.cogentex.com/solutions/fogindex.shtml

where *N* is a finite set of non-terminal symbols, *Te* is a finite set of terminal symbols for the input language, *Tf* is a finite set of terminal symbols for the output language, *L* is a lexicon consisting of a finite set of production rules, and $S \in N$ is a distinguished start symbol. Grammars are either constructed manually or learned from an aligned parallel corpus of utterances in a natural language and their respective semantic realizations, like in (Wong and Mooney, 2007) (see Fig.2.9).



R1 => [(bowner TEAM_1 {UNUM_2 }) , TEAM_1 player UNUM_2 has the ball]

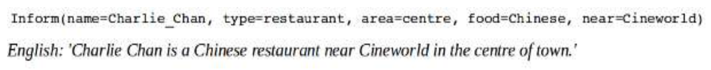
TEAM_1 => [our, 'our']

UNUM_2 => [6, '6']

Fig. 2.9 Production rules from Wong's grammar yielding the sentence *'Our player 6 has the ball.'*

Wong uses Synchronous Context-Free Grammars (SCFG) as unique semantic formalism for both NL generation and understanding. Given that an SCFG is fully symmetric with respect to both generated strings, it can indeed serve as the core semantic formalism for both NLG and NLU.

6. **Stacks of concept/value pairs**. (Mairesse et al., 2010) uses stack-based semantic formalism, where each dialog act is represented by a dialogue act type (bottom concept framing the entire stack) and an unordered set of concepts along with their values, like in Fig. 2.10.



```
Inform(name=Charlie_Chan, type=restaurant, area=centre, food=Chinese, near=Cineworld)
```
English: 'Charlie Chan is a Chinese restaurant near Cineworld in the centre of town.'

Fig. 2.10 Concept stack representation of a dialog act used by F. Mairesse

Here $inform$ is the dialog act type framing the entire stack and defining the subsequent utterance type, and *name=Charlie_Chan, type=restaurant*, etc.represent semantic concepts along with their respective lexical values, which form the main content of the dialog act. This formalism is used by our generation systems (see chapter 3).

7. **Linked data and Resource Description Framework (RDF)**. RDF uses a graph-based data model for knowledge representation. Statements in RDF are expressed as so-called triples of the form *'subject : predicate : object'*, where predicate is a binary relation taking subject and object as arguments. RDF subjects and predicates are Uniform Resource Identifiers (URIs) and objects are either URIs or literals. For example, the RDF statements in Fig.2.11 are intended to express the following: *'the J. S. Bach's date of birth is 1685-03-21 and his place of birth is Eisenach'*.

```
:Johann_Sebastian_Bach dbont:birthDate "1685-03-21" .
:Johann_Sebastian_Bach dbont:birthPlace :Eisenach .
```

Fig. 2.11 RDF format

(Duma and Klein, 2013) uses a parallel corpus of RDF forms and their lexicalized counterparts to extract generation templates (Fig.2.12).

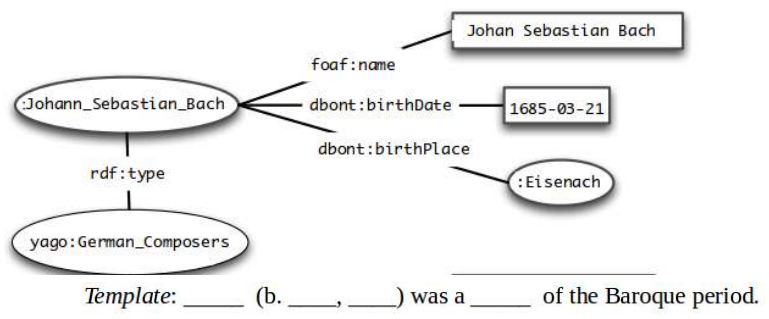


Fig. 2.12 RDF net representation

As we mentioned earlier despite a huge number and a wide variety of existing formalisms, new ones continue to appear. None of them is considered to be universal and general enough to be applicable to any NLG system. Thus developing a particular semantic formalism often constitutes a part of NLG system development process.

2.1.4 NLG paradigms

Beside handling different output modes (spoken VS written) and using different semantic formalisms, NLG systems differ in the generation paradigms used by their core generation models. Based on their underlying model NLG systems can be divided into rule-based, semi-data-driven and fully data-driven. Each of these groups has its strong and weak points. We discuss each of these groups in detail in the following subsections.

Rule-based models

This group includes *predefined (canned) text*, *template-based* and *grammar-based* models. These models are widely (and primarily) used in written NLG, where such characteristics as precision, robustness and uniformness of the output are highly valued.

Predefined-text model is the simplest (in terms of design) NLG model in which the output is limited to a set of utterances preliminary designed for the system. An example of such a model is **NPCEditor** (Leuski and Traum, 2010) which allows to build spoken dialog agents with predefined set of dialog acts / responses.

Template-based models use manually constructed sets of templates to generate an output phrase. A template is a predefined model phrase in a natural language which corresponds to a particular statement in a formal language. Each time a given statement is met it is replaced with a corresponding template to produce the output phrase. Templates can contain variables which most often serve as slots for variable values in formal statements.

Grammar-based models as their name suggests use hand-crafted grammars and vocabulary to generate the output phrases. The difference between a grammar rule and a template is that a grammar rule can generate different templates using various vocabulary units. Thus a grammar has a somewhat larger «generative power» and can be used in tasks that do not require expression rigor and allow for more flexibility in their output.

Building and maintaining both template-based and grammar-based models require extensive manual work on the part of the developers. Among most well-known examples of rule-based generation systems are **Cosma** Appointment scheduling agent (Busemann et al., 1997), **PENMAN** (Bateman et al., 1990) and a more recent and more complex **SmartKom** system (Wahlster, 2006). Examples of grammar-based systems: **RealPro** (Lavoie and Rambow, 1997) and **FUF/SURGE** (Elhadad et al., 1997).

Until recently spoken NLG systems widely used template-based models. **YAG** (Channarukul et al., 2001) uses declarative, template-based approach for the representation of text structure. YAG relies on a relatively small number of templates and a powerful description language to provide a fine-grained sentence specification. **EXEMPLARS** (White and Caldwell, 1998) is a rule-based, object-oriented framework for dynamic generation of text and hypertext. The EXEMPLARS framework allows for combination of phrasal templates and sophisticated linguistic models . The basis of the framework is the notion of an *exemplar*, a template-like text planning rule. Unlike traditional textual templates, which tend to produce rigid text, exemplars are dynamic[7], which makes it easy to generate a variety of texts, in response to different input conditions.

Though rule-based generation models were ideal for written NLG within limited domains and tasks, they were somewhat too restrictive for spoken NLG. At the same time, models allowing for more freedom and flexibility in their output were error-prone and needed additional output validation which was too costly to perform. With time NLG systems started to gradually evolve in the direction of data-driven learning.

Semi-data-driven models

First there were attempts to introduce isolated statistical components into rule-based NLG systems. Particularly, n-gram language models (LM) were used to help the generation process

[7]Exemplars being objects, they can be programmed to allow for recursion, dynamic clause generation, etc.

by picking the most likely variant of the output phrase. Thus **NITROGEN** (Langkilde and Knight, 1998) and **HALOGEN** (Langkilde-Geary, 2002) are among those systems which use n-gram LMs for re-ranking of the output phrases. The core generation models remained rule-based though.

FERGUS (Bangalore and Rambow, 2000) was among the first NLG engines having statistical learning at its core. It builds a word lattice from a dependency tree using a statistical tree model. Then the n-gram LM is used to find the best scoring output variants.

AMALGAM (Corston-Oliver et al., 2002) is an NLG system which performs dependency parsing using decision tree classifiers. These classifiers are also used for local re-orderings, insertion of auxiliary words, etc. Large-scale re-ordering is performed using an n-gram LM. The tree classifiers are trained on the output of manually constructed parsers.

Although these models use limited (local) statistics-based elements, they have been a huge step forward in transition to purely data-driven models and towards creating more flexible and extensible NLG systems. Many of the ideas introduced by their developers are widely used in modern NLG frameworks.

Pure data-driven models

As our work revolves around spoken language generation within the framework of a dialog system, we focus on the recent advances in this area of NLG. For developments in statistical written language generation see (Angeli et al., 2010; Banko et al., 2000; Belz, 2008; Turner et al., 2006).

Purely statistical NLG models started to appear quite recently. Yet a wide spectrum of models and techniques have been proposed and many of them showed similar or better performance than their rule-based counterparts. The idea of building data-driven NLG systems proved to be viable and continues to attract attention of researches.

For presentation convenience, we divide here all statistical NLG models into two groups: SMT-based models and other learning-based models. The later category includes a range of different statistical models. Yet they all share a major common feature: they are all learned from a training corpus, just like SMT models are. In the following section we provide an overview of data-driven NLG models, starting with SMT models, all the way up to the most recent learning-based techniques.

2.1.5 Translation-based and other data-driven NLG models

Statistical machine translation (SMT) is a well established field with a variety of approaches and available tools (see section 2.2.1). As we discussed earlier SMT proposes some interesting properties that might be beneficial for NLG: extraction of smaller reusable units, a particular reordering scheme, target language model and a number of others. Each SMT paradigm offers a specific set of features, and that in turn might determine the choice of a particular SMT model.

Phrase-based models

Phrase-based SMT is a state-of-the-art machine translation paradigm. It offers some interesting characteristics which may be beneficial for NLG, specifically extraction of a translation/generation dictionary in a form of a phrase-table, basic reordering, a target language model, an easy combination and integration of feature functions in a log-linear model as well as tuning of their parameters, beam-search decoding. The main drawback of these models is their relatively week reordering heuristics which fails to provide a robust reordering scheme in the generation scenario, where often the semantic units in the source side are not ordered.

Also their scoring functions based mostly on normalized maximum likelihood estimations do not always provide reliable model estimations due to the fact that the generation units (pairs of semantic and lexical items, extracted in accordance with the alignment) are scored in isolation (out of context). Finally there is no handling of OOV concepts, which is tolerable in some cases in SMT (like in cases of proper names and numbers) but is unacceptable in NLG where the source side represents a semantic formalism non-intelligible for average users. A typical example of such a model is **MOUNTAIN** generation system (Langner et al., 2010) – an NLG system based on classical phrase-based SMT model.

Syntax-based models

Syntactic models for NLG have been employed in a number of studies. Just like in syntax-based SMT, they offer a framework for language generation which uses chart decoding to produce an output phrase. Unlike phrase-based SMT, syntax-based models handle long-distance reordering much better. Moreover they are able to model not contiguous suits of both semantic and lexical units (words), which makes them more robust in handling complex semantic formalism that use nested concepts. Finally chart decoding presents a viable alternative to beam search decoding.

Two most well-known systems built using syntax-based SMT paradigm are **WASP-1** by H. Wong and a **tree CRF-based** NLG system by Y. Lu.

(Lu et al., 2009) introduced a tree CRF-based model. It uses a hybrid tree formalism that combines both natural language sentences and their meaning representation in a tree structure. Then tree conditional random fields are used on top of the hybrid tree representation, which allows for an explicit modeling of phrase-level dependencies between neighboring natural language phrases and meaning representation components. The authors claim that their model outperforms WASP-1 in terms of the BLEU score (for the definition of BLEU see section 2.2.6).

Notably, both WASP-1 and Lu's tree CRF-based model represent reversed SLU algorithms; in both cases one framework is used for both semantic tagging and NL generation. Syntax-based NLG does solve certain problems which are poorly handled by phrase-based NLG, like long distance reordering and embedded concepts in the source side. The main difficulty in building such models is the need for additional annotation of the training corpus in the form of syntactic parsing, morphological categorization, etc, which is costly to produce manually and which might introduce errors when obtained using automatic tools.

N-gram Translation Models

This MT paradigm was used (as one of the components) by (Oh and Rudnicky, 2000) to build the generation module in the Carnegie Mellon Communicator - a telephone-based spoken dialog system serving to help the users arrange their travel itineraries (Rudnicky et al., 1999).

Oh's NLG system generates a number of generic utterances using 5-gram model built using the corpus of in-domain dialogues, annotated with semantic categories used by the in-house dialog manager (word and utterance classes). The corpus is pre-processed so that all named entities and other informative words are replaced with generic class names. Thus the semantic representation of a dialogue act provided by the DM serves as the base for generating utterances of specific class. The produced utterances are scored according to several criteria (the length, repetitions, etc) and the best one is selected. Then the generic class words are replaced with actual values supplied by the DM to obtain the final output sentence.

In our study we continue along this line of research and use n-gram SMT paradigm to build a generation model. But unlike Oh's, our n-gram model is integrated into a larger generation system which combines n-gram and CRF generation models, a reordering model and a target language model. To our knowledge there has been no attempts to build an NLG system using a complete n-gram SMT pipeline (including reordering and target LM). As we observed during our experiments (see section 3.4), n-gram models show a significantly better performance on our data than a classical phrase-based SMT model.

Other data-driven generation models

Recently a number of studies have been carried out on purely statistical modeling in NLG. These models do not use any hand-crafted rules or grammars; the core generation model is either learned from an aligned parallel corpus or derived from a large monolingual data-set via the application of specific constraints.

Thus (Mairesse et al., 2010) uses Dynamic Bayesian Networks to map semantic representation to a natural language utterance. The junction tree algorithm is then employed to search for the highest scoring sequence.

(Dinu and Baroni, 2014) suggests that the generation can be performed using distributional semantics models. This method is not conceived specifically for spoken language generation, but considering the novelty of the approach, its potential and its fitness to the task of spoken NLG, we think it useful to mention it here.

(Swanson et al., 2014) proposes to use constraints in structure instead of semantic constrains (conceptual content of a phrase) while generating a phrase. Thus the generation is performed by searching for a sentence in a corpus which meets these constrains (contains certain words). Swanson's study actually deals with free-form data driven NLG, but the methodology looks promising and applicable to dialog system NLG as well.

(Wen et al., 2015) develops a statistical language generator based on a joint recurrent and convolutional neural network structure. The authors claim that their model outperforms n-gram and rule-based generators under the same experimental conditions.

2.1.6 Learning grammar/templates from a corpus

Building semantic grammars and generation templates require extensive manual work on the part of system developers and domain experts. There have been attempts to reduce the amount of work by automating theses tasks by searching for templates and grammar rules in a corpus. This idea is behind the so-called example-based NLG (DeVault et al., 2008): learning the generation grammar for a particular semantic formalism from a corpus of training examples. The authors adopt the probabilistic tree-adjoining grammar formalism and grammar induction technique of D.Chiang (Chiang, 2005) in order to produce grammar derivations from parsed and semantically annotated corpus.

There have also been a number of studies on learning semantic grammars from a corpus. For example, (Kate et al., 2005) suggests learning transformation rules from a set of natural language sentences paired with their formal representations and then use these rules to perform the mapping between the two.

Crowd sourcing has been gaining more popularity as a methodology for obtaining templates for template-based NLG, particularly with the advent of tools like Amazon's Mechanical Turk[8] which allow to collect information from many people fast and at a low cost.

(Mitchell et al., 2014) uses crowd sourcing to collect new generation templates which are then validated by system developers. Thus the annotators were presented with a dialog act and were asked to produce new phrases for particular parts (utterances) of that act. These templates were then used in a template-base system. The authors suggest that gathering language templates from multiple untrained annotators, as opposed to a few experts or system developers may help produce a more natural distribution of alternative phrasings in a dialogue.

(a) [You]$_{\text{agent}}$ [have not finished]$_{\text{action}}$ [your thesis]$_{\text{patient}}$

(b) [The students]$_{\text{agent}}$ [have not finished]$_{\text{action}}$ [their homework]$_{\text{patient}}$

(c) [The students]$_{\text{agent}}$ [will complete]$_{\text{action}}$ [their homework]$_{\text{patient}}$

Fig. 2.13 Corpus template (a) and newly generated sentences (b), (c) Tadeau, 2010

(Tadeu et al., 2010) propose to extract generation templates from a corpus of in-domain (or similar) texts. They perform semantic role labeling and syntactic parsing in order to obtain 'replaceable' parts of a phrase; they select a sentence that resembles the desired output, and then replace some or all of its constituents according to the output semantic specification (Fig.2.13). This method was initially conceived for template-based systems, but it might be a good start for creating a set of basic templates in a data-driven approach, which might then be extended automatically, e. g. with paraphrasing.

(Duma and Klein, 2013) used Wikipedia and Resource Description Framework (RDF) to find and retrieve generation templates (see Fig.2.12).

2.1.7 Adaptive natural language generation (user adaptation)

As we pointed out earlier one of the major drawbacks of template-based and grammar-based NLG systems is their inability to adapt to generation context changes and user preferences. As the set of templates and the number of possible structures generated by a grammar are limited in number, the output generated in different meta-linguistic scenarios would be the

[8]www.mturk.com

same. In a rule-based systems the adaptation option might be included via hard-coding, just like the core generation engine. Trainable (data-driven) NLG naturally allows for inclusion of an adaptation capacity which might be learned along with the content selection, content ordering, etc. (Lemon, 2008) proposes to model the NLG problem within a dialog system as a Markov Decision Process, specifically the author models the decision of how to present a list of items to a user in an MDR and solves it by trial-and-error exploration, using Reinforcement Learning. According to the author, this adaptive NLG policy shows a 27% relative increase in reward over a baseline policy for the same task.

When it comes to generating the output for different user groups, several methods have been proposed, such as utilizing different system parameters for different user groups (Mahamood and Reiter, 2011) and building user models (Janarthanam and Lemon, 2010). A somewhat different approach has been described in (Gkatzia et al., 2014). The author presents a multi-adaptive summary generation system able to adapt the summaries produced for either of 2 groups of users: students and professors. It uses knowledge derived from ratings on feedback summaries by extracting the most relevant features using Principal Component Regression (PCR) analysis, in order to identify the users' preferences. These are then included into a reward function, that is used for training the Reinforcement Learning agents which learn to make optimal content selection decisions (either adapting to professors, to students, or to both groups simultaneously).

2.1.8 Corpora collection for data-driven NLG

Unlike in machine translation domain where there exist a number of ready-to-use bilingual resources (like Europarl (Koehn et al., 2002), multilingual News Commentary corpora, etc), NLG has much less readily available training data of that kind. Having an abstract semantic (and in essence artificial) language on one side, the training data cannot be obtained from simply recording human interactions and normalizing/aligning the produced corpus; it needs to be created manually, at least when it comes to semantic annotation.

Moreover a variety of semantic formalisms and narrow domain specificity makes it impossible to join all the available corpora, which are themselves very small due to the construction cost, into a unique and unified corpus of a substantial size. Normally, the construction of such a corpus requires manual annotation and sometimes also alignment between semantic and surface realization (like in (Mitchell et al., 2014)).

As data-driven models proved their robustness and established their position among the best-performing NLG paradigms, the attention has been shifting to low-cost training corpus acquisition. This problem has been addressed in a number of recent studies. Most of them resort to crowd-sourcing as their principal corpus collection method.

(Mitchell et al., 2014) and (Tadeu et al., 2010) that we mentioned above suggested their methodology for extracting templates for template-based systems. But these methods are also perfectly suited for building the training corpus in a data-driven scenario, if we regard the obtained utterances and their respective semantic realizations as an annotated parallel training corpus.

(Mairesse et al., 2010) followed the approach similar to (Mitchell et al., 2014), but instead of presenting the annotators with an example phrase to be replaced, they were provided with semantic representations of phrases in the form of concept stacks and asked to come up with phrases delivering all the information contained in those concept stacks. The collected corpus of utterances along with their semantic realizations was used to train their BAGEL generation model (see section 2.1.4).

The most frequently used and reliable method of corpus acquisition remains manual semantic annotation of in-domain texts by expert annotators. (Reiter et al., 2003) gives an overview of manual corpus collection techniques, including corpus-based data collection and expert annotations; he denotes that expert-produced corpus may be ’one-sided’ and may not cover sufficiently all the necessary phenomena, while corpus-based approach may not be accurate enough in terms of distinguishing between different phenomena. Finally he highlights the importance of mixing different techniques to obtain a balanced and diverse corpus, and also the importance of subsequent validation.

2.1.9 NLG evaluation

Evaluation is an important step in the development of any NLP application. But unlike the applications with a precise and well-defined output, like POS tagging, named entities recognition, and the like, systems dealing with natural language production in some way (machine translation, text summarization, natural language generation, etc.) are difficult to evaluate due to the subjective nature of judgments on their quality and also their ability to produce multiple outputs which cannot be matched against a single reference. This ability restricts the use of automatic metrics to the development period and makes them barely suitable for final evaluation. In addition, as generation systems produce the output intended to be interpreted by humans, there is a number of additional criteria for evaluation. Apart from purely quantitative scores like accuracy and speed of processing, we would also like to evaluate fluency, clarity, ’humanness’ and diversity, which are hard to measure/quantify and the judgments on these criteria are highly subjective. As a generation module is usually a part of a larger dialog system, two types of evaluation can be performed : intrinsic and extrinsic.

Intrinsic evaluation assesses the properties of systems in isolation from the application context, e.g. comparing their outputs to one or more references or analyzing user feedback. This type of evaluation is often less expensive and more informative as it is free from external errors produced by other modules or components of a larger system, that might interfere during evaluation process.

Extrinsic evaluation assesses the performance of the module within a larger system or context (e.g. generation module within a dialog system) and its effect on the communication process in general (success of the interaction). Extrinsic evaluation is more costly and more difficult to set up and carry out, but at the same time it is indispensable as it allows to assess the impact of the NLG module performance on the whole system.

A generation system can be evaluated either by human judges or using an automatic metric, borrowed from a related NLP area, like e.g. SMT.

Human evaluation

Human evaluation is the process of assessing the quality of system outputs according to different quality criteria, performed by either trained evaluators or ordinary users. Typically it is done using rating scales, but it can also represent human assessment of the degree of similarity between system outputs and reference outputs. Evaluation may be comparative or absolute. In case of comparative evaluation users are normally asked to compare several outputs of the same or of different systems and to rank them according to their preference or to a given criterion (fluency, clarity, etc). In an absolute evaluation scenario the user is presented with the output of a single system and either gives it a score (within some scale, e. g. 1-5) or in case of binary selection, marks the output as acceptable or not. The typical criteria for human evaluation are:

- **adequacy** – a judgment on how well the generated phrase transmits the meaning intended by the dialog manager; specifically it checks if all information has been presented without any distortion;
- **fluency** is a judgment on how well-formed and natural the produced phrases are; it does not take into consideration the information contained in a phrase, but rather its overall grammaticalness and correctness;
- **success** – a binary metric consisting of accepting or rejecting the instance based on general satisfaction of its quality;

Some other criteria that might be taken into consideration when evaluating an NLG system are **clarity, humanness** and **diversity**.

Automatic evaluation

Human evaluation, despite being the most reliable type of assessment, is very costly to perform, especially during the development period, when it is necessary to run quick tests in order to evaluate the impact of minor changes in system parameters.

Automatic metrics are actively used in the MT research community and have been shown to have sufficient correlation with human judgments. The proximity of the tasks of MT and NLG in terms of their output format made it possible for NLG to use the same evaluation metrics. Automatic evaluation allows to assess the performance of a system quickly and at no cost. It normally consists in calculating the degree of similarity between system outputs and one or more reference phrases. We discuss automatic metrics in detail in section 2.2.6. Here we will briefly outline the most widely used ones:

- **BLEU** score is an n-gram based string comparison;
- **NIST-5** is a weighted version of BLEU, with more importance given to less frequent n-grams;
- **ROUGE** in its original variant (ROUGE-N) represents n-gram co-occurrence statistics; there are variants for longest co-occurring subsequences (ROUGE-L) and skip-bigram co-occurrence statistics (ROUGE-S);

Also there exists a number of metrics which are not used that often but might be of interest in specific evaluation setups:

- **accuracy** is a proportion of outputs that are identical to the corresponding human description or reference;
- **string edit (Levenshtein) distance** is a proportion of insertions, deletions and substitutions required to convert a system output into an acceptable phrase.

The performance of a particular evaluation metric is estimated in terms of how well it correlates with human judgments on the quality of system outputs. The correlation estimates are obtained using Pearson product-moment correlation coefficient or Spearman rank-order correlation coefficient (Pearson, 1895).

There is an ongoing discussion about the drawbacks and advantages of automatic evaluation in areas like MT and NLG. Thus it is believed that strict corpus-similarity measures tend to favor repetitive generation strategies that do not diverge much, on average, from the corpus data, while human judges often prefer output with more variety (Foster, 2008; Mairesse et al., 2010), even if the output is not perfectly fluent.

A study carried out by Anja Belz (Belz and Reiter, 2006) shows that out of NIST, BLEU and ROUGE, NIST scores correlate best with human judgments and that all of them are biased in favor of generators that select the output on the basis of frequency alone. The author also suggests that the number of required references for proper NLG system evaluation is greater than that for MT, as the variation scale is larger and the output can be very different from the reference, but still effectively meet the system's communicative objective. A. Belz also denotes that ROUGE is the least reliable metric and it does not seem to offer any advantage over simple string-edit distance. Another interesting conclusion that the author comes to as a result of her study is that individual experts' judgments are not likely to correlate highly with average expert opinion, even less likely than NIST scores. Thus if expert evaluation can only be done with one or two experts, it is less likely to be as reliable as a NIST-based evaluation performed on a high-quality reference corpus produced by multiple different authors and offering sufficient diversity and variation.

Despite a relatively good performance of some automatic metrics (like NIST and BLEU), human evaluation remains the most widely used form of NLG output validation and as a rule concludes the development of most NLG systems.

2.2 Statistical Machine Translation

In this section we provide theoretical background of the methods and techniques that we used to build our NLG systems. We start with an introduction to statistical machine translation (section 2.2.1) and its core components: alignment, reordering, translation models and target language models (2.2.2). We take a closer look at the translation model and various SMT paradigms in corresponding subsections of section 2.2.3: phrase-based, syntax-based, factored SMT which is a variation of phrase-based models and finally n-gram SMT framework. In section 2.2.4 we present the minimum error rate training algorithm, which is generally used for learning the optimal parameters in log-linear models. In section 2.2.4 we discuss discriminative learning (DL) in SMT, specifically the DL model that we use in our NLG system, namely Conditional Random Fields. Finally we end this part with the section on MT evaluation (section 2.2.6).

2.2.1 SMT frameworks

Machine translation is the area of computational linguistics that is concerned with developing computer software which serves to translate text or speech from one language to another.

Several categories of MT systems can be distinguished, based on their architecture and how they generate the output.

Rule-based MT systems encode linguistic information about source and target language grammars into translation rules and use them to produce the output.

Example-based MT searches its knowledge base (bilingual corpora) for the most likely (most similar) translation of a given input at run-time.

Hybrid MT systems combines multiple translation models (of different nature) within a single machine translation system.

Statistical MT is a machine translation paradigm where translations are generated according to statistical models trained on bilingual (parallel) text corpora. Several different SMT paradigms have been developed over the last twenty years. In the following subsection we take a closer look at the state-of-the-art SMT framework.

State-of-the-art SMT models

First experiments in SMT which date back to late 80s early 90s used **word-based SMT models** ((Brown et al., 1990)): the translation of a sentence represented the combination of lexical translations of words composing this sentence. Thus the context was not taken into consideration while translating a particular word. The possible translations for each word along with their probability distributions were collected from a parallel corpus, aligned at the sentence-level. The limitations of such models soon became apparent and **phrase-based SMT models** (Koehn et al., 2003) were introduced. These models learn along with lexical translations of words the translations of contiguous phrases which allows for translation of multi-word expressions or even short sentences – something that was not possible with word-based models. These models are still considered to be the most robust and efficient and are widely used in the commercial sector as well as in the research community.

Training an SMT model consists in retrieving bilingual phrases consistent with word alignment from a parallel corpus. Each phrase is assigned a translation probability - calculated as a simple maximum likelihood estimation - and a number of additional scores which may vary from one model to another. The standard set of additional scores includes direct and reverse lexical scores (word-by-word translation probabilities within a phrase estimated using maximum likelihood principle), reverse phrase translation (source and target are reversed) and sometimes lexicalized reordering scores. The final phrase translation score represents a linear combination of interpolated smaller scores. When translating an input sentence using a phrase-based model the translation hypothesis score is calculated as follows:

$$p(f_1^I|e_1^I) = \prod_{i=1}^{I} \phi(f_1^I|e_1^I) d(start_i - end_{i-1} - 1) \tag{2.2}$$

where ϕ is the translation score and d is the reordering score calculated given the start and the end index of a given phrase. Finally the product is combined with a language model score to produce an overall hypothesis result. This model is known as a noisy channel approach (Fig. 2.3).

$$e_{best} = argmax_e(p(e|f)) = argmax_e(p(f|e)) * p_{LM}(e) \tag{2.3}$$

Some other features might be added to the final score estimation: distortion penalty which adds a 'cost' for each reordering skip; word penalty which ensures that the translations do not get too long or too short, unknown word penalty, etc.

The hypothesis with the highest overall score is selected as «the best» translation. The selection process is performed during **beam-search decoding**, when an input sentence is decomposed into parts and partial score for each hypothesis is calculated, thus allowing to abandon bad (low-scoring) hypotheses early on, which decreases the decoding time and computational complexity. This process is called hypotheses pruning and is performed either by setting a threshold on the partial hypothesis score or by setting a limit to the number of produced hypothesis.

Target language model (LM) is an important component of an SMT model as it ensures the fluency of the output and helps to select the most plausible translation, which might not be the best one according to the translation model. It represents an n-gram model trained on a large monolingual target corpus. We discuss language modeling in section 2.2.2 below.

Another indispensable component of an SMT system is the **reordering model**. The actual model currently used in phrase-based models is fairly weak; it implements a simple heuristic which punishes long distance movements. This heuristic is not able to model long-distance syntactically-motivated movements, like moving the main verb to the end of a phrase in Japanese. This problem was partially solved with the advent of syntactic SMT models (see subsection 'Syntax-based translation models' below). The problem of reordering is an important issue not only in MT but also in NLG, regardless of the core generation model. We discuss this issue in section 2.2.2.

Log-linear models

A standard SMT paradigm represents a weighed combination of feature functions (models). Three basic components of any SMT system are translation model, language model and reordering model. Other model components may include lexicalized reordering scores, lexical translation scores, word penalty, etc. These model components (along with a weight for each) are joined into a log-linear model (Fig. **??**), which is used to calculate hypotheses scores during decoding and to select the best translation. The weights λ can be adjusted to maximize a particular evaluation metric. The common way of tuning the weights is the Minimum Error Rate training (Och, 2003) (see section 2.2.4).

$$p(x) = exp \sum_{i=1}^{n} \lambda_i h_i(x) \tag{2.4}$$

$$\begin{aligned} p(e,a|f) = exp(\lambda_{\omega} \sum_{i=1}^{I} log\ \omega(f_i|e_i) + \\ \lambda_d \sum_{i=1}^{I} logd(a_i - b_{i-1} - 1) + \\ \lambda_{LM} \sum_{i=1}^{|e|} logp_{LM}(e_i|e_1...e_{i-1})) \end{aligned} \tag{2.5}$$

Fig. 2.14 Log-linear model

2.2.2 Model components

In this section we take a closer look at three indispensable SMT model components: alignment, reordering and target language modelling.

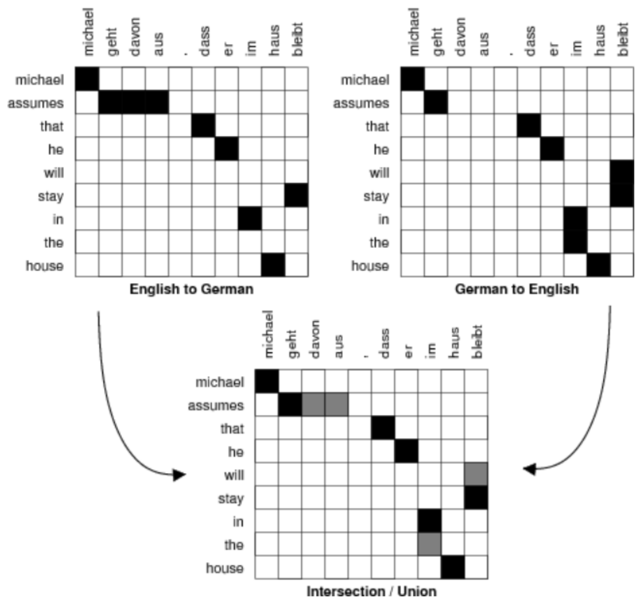
Alignment and phrase extraction

The first step in building the model in corpus-based translation/generation paradigms is creating a dictionary of small reusable units from the parallel corpus aligned at the word level. Alignment is the process of identifying translation equivalents among the words in a sentence-aligned parallel corpus[9].

[9] Word alignment is typically done after sentence alignment has already identified pairs of sentences that are translations of one another.

The alignment model is trained using the IBM models (Brown et al., 1990). These models are based on the Expectation–maximization algorithm which represents an iterative method for finding maximum likelihood or maximum a posteriori estimates of parameters in statistical models that use hidden (latent) variables. During the expectation-step the translation probabilities within each sentence are computed; this step creates a function for the expectation of the log-likelihood evaluated using the current estimate for the parameters. During the maximization step the translation probabilities at the sentence level are accumulated to global translation probabilities. The maximization step computes parameters maximizing the expected log-likelihood found during the expectation step. These parameter-estimates are then used to determine the distribution of the latent variables in the next expectation step. The two steps are repeated until conversion.

Fig. 2.15 Alignment of the German sentence 'Michael geht davon aus , das er im Haus bleibt' and its English equivalent 'Michael assumes that he will stay in the house'. Picture from http://www.statmt.org/moses/.



The alignment process within the translation/generation pipeline unfolds as follows: first, the parallel corpus is aligned in both directions (source-target and target-source). This generates two word alignments that have to be reconciled: the intersection of the two alignments produces a high-precision alignment of high-confidence alignment points; the union of the two alignments produces a high-recall alignment with more additional alignment points. Other alignment schemes include using only source-to-target or only target-to-source alignment points with the possibility of including some additional points from the union. The choice for a particular alignment scheme depends on the task at hand.

Reordering

Reordering is one of the biggest challenges in MT and also in data-driven NLG, especially if there are several distinct semantic units in the input and they are not ordered. Different MT paradigms tackle differently the problem of reordering inside the translation models. Phrase-based paradigm performs reordering while decoding; reordering is distance-based, it punishes long-distance movements by assigning a higher cost to longer skips. Scoring function:

$$d(x) = a^{|x|}$$

is exponential with skipped distance. This heuristic is crude and does not model explicitly reordering patterns, nor captures regularities in the source-target mapping; it does not make use of training examples and it is not linguistically motivated. The reordering inconsistencies in the phrase-based approach are supposed to be balanced by a target LM. In summary, phrase-based systems have relatively limited potential to model word-order differences between different languages.

Several other reordering solutions have been proposed for SMT systems. (Collins et al., 2005) suggests to parse the input string to obtain a syntactic constituents of the sentence and apply hand-crafted reordering rules as a pre-processing step to copy the order in the target. (Xia and McCord, 2004) exploits the same idea, but instead of hand-crafted rules he proposes to use automatically learned rewrite patterns (which are linguistically motivated, as they are obtained by parsing the two sides) for pre-process the source sentence so that the word order is similar to the one in the target.

As we already discussed in subsection 2.2.3 above, (Crego and Mariño, 2006) suggests to couple reordering and decoding; reordering model represents linguistically motivated rewrite rules, which are learned from the training corpus and used to extend a monotonic search graph with reordering hypotheses. The extended graph is traversed in the global search to find the best scoring output sentence.

In data-driven NLG the problem of reordering has not been studied thoroughly. In many cases it is not handled in any specific way. In BAGEL (Mairesse et al., 2010) there is no explicit reordering modeling and the decoding algorithm considers all possible orderings together with all possible lexical realizations. In WASP-1 (Wong and Mooney, 2007) the input semantic concepts are pre-ordered, so there is no need for a separate reordering model. Thus reordering remains one of the 'open problems' in the NLG domain.

Language models

A statistical language model (LM) estimates the relative likelihood of a sequence of words $P(w_1,\ldots,w_n)$ based on the probability distributions in the training corpus. Basically it indicates how likely a given sequence is to occur in the corpus, or more broadly in that language. Language models are used in many natural language processing applications: speech recognition, machine translation,text summarization, language generation, handwriting recognition and other applications.

Fig. 2.16 N-gram language model

$$p(w_1,...,w_m) = \prod_{i=1}^{m} p(w_i|w_1,...,w_{i-1}) \approx \prod_{i=1}^{m} p(w_i|w_{i-(n-1)},...,w_{i-1}) \tag{2.6}$$

It is assumed that the probability of observing the i^{th} word w_i in the context history of the preceding $i-1$ words can be approximated by the probability of observing it in the context history of the preceding $n-1$ words (Fig.2.16).

LMs quality can be measured with cross-entropy (2.7) or perplexity (2.8)

Fig. 2.17 Cross-entropy and perplexity

$$H(SENT) = \frac{1}{n} logp(SENT_1^n) \tag{2.7}$$

$$Perplexity(SENT) = 2^{H(SENT)} \tag{2.8}$$

2.2.3 Other SMT paradigms

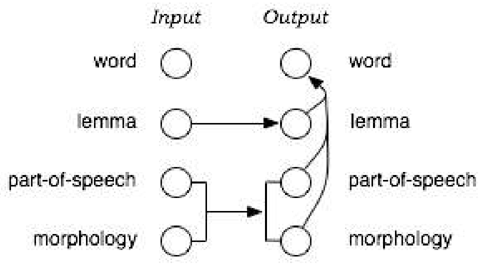
In section 2.2.1 we discussed the state-of-the art translation models which are widely used in both research and commercial sector today. Despite the general accord on the robustness and relatively good performance of the phrase-based SMT framework, a number of alternative translation paradigms have been developed; among them are n-gram and syntax-based SMT models. In the following subsections we present these frameworks; we also take a quick look at factored translation models which are an extension of the phrase-based paradigm.

Factored translation models

Neither word-based nor phrase-based models use any kind of linguistic knowledge; both rely solely on statistical calculations performed on the training data. Nevertheless linguistic information like morphological categories (parts of speech, number, case, etc), semantic tags (word classes, semantic roles, etc.), and other kinds of linguistic annotation in the training corpus (named entities, anaphoras, etc.) were shown to help the translation process and improve the quality of the output ((Banchs and Costa-jussà, 2011; Carpuat, 2009; Wu and Fung, 2009)), especially for morphologically rich and resource-scarce languages (Ceausu, 2011).

There exist several ways to integrate this information into the translation model; the most common one is building **factored models** (Koehn and Hoang, 2007). They allow for inclusion of linguistic information such as POS, morphological categories, etc. by adding the tag containing a given linguistic category as a factor to each word (see Fig.2.18).

Fig. 2.18 Factored training. Picture from www.statmt.org

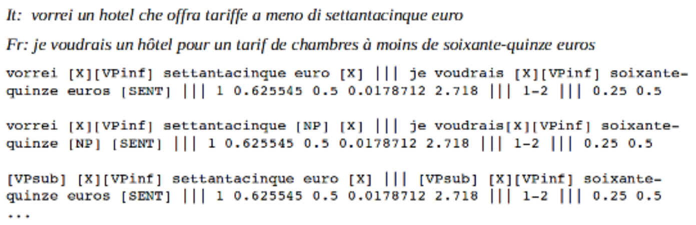


During training a factor pattern is specified, determining the mapping between particular factors in the source and the target sides. Adding factors to the source side might help reduce the ambiguity in translation selection; factors on the target side are used in a separate generation step, in order to generate the final translation from target factors, or as the base for training factor target language models.

Syntax-based translation models

Syntactic translation models were first introduced by (Yamada and Knight, 2001), then later developed by (Chiang, 2005), (Nesson et al., 2006) and (Zollmann and Venugopal, 2006). These models are based on synchronous grammars that are learned from parallel corpus in the same way as phrase-based models, i.e. via extracting source-target phrase couples consistent with the parse trees alignment. The couples are converted into grammar rules by replacing contiguous subphrases with either a generic non-terminal X, or, if the corpus is annotated with syntactic information, an actual syntactic subtree label (both in accordance with the inner syntactic representation).

The major advantage of these models is their ability to learn long distance reordering schemes as well as non-contiguous phrase translations.



It: vorrei un hotel che offra tariffe a meno di settantacinque euro

Fr: je voudrais un hôtel pour un tarif de chambres à moins de soixante-quinze euros

```
vorrei [X][VPinf] settantacinque euro [X] ||| je voudrais [X][VPinf] soixante-
quinze euros [SENT] ||| 1 0.625545 0.5 0.0178712 2.718 ||| 1-2 ||| 0.25 0.5

vorrei [X][VPinf] settantacinque [NP] [X] ||| je voudrais[X][VPinf] soixante-
quinze [NP] [SENT] ||| 1 0.625545 0.5 0.0178712 2.718 ||| 1-2 ||| 0.25 0.5

[VPsub] [X][VPinf] settantacinque euro [X] ||| [VPsub] [X][VPinf] soixante-
quinze euros [SENT] ||| 1 0.625545 0.5 0.0178712 2.718 ||| 1-2 ||| 0.25 0.5
...
```

Fig. 2.19 Entry from a syntax-based translation (Italian-French)

N-gram translation models

This approach to machine translation, proposed by (Marino et al., 2006), and developed by (Crego and Marino, 2007), differs significantly from the traditional phrase-based models in many respects. Here the translation model is based on n-gram scoring of bilingual units called tuples. Tuples resemble phrase pairs in the phrase-based approach; they represent a coupling between given source and target word sequences consistent with word alignment.

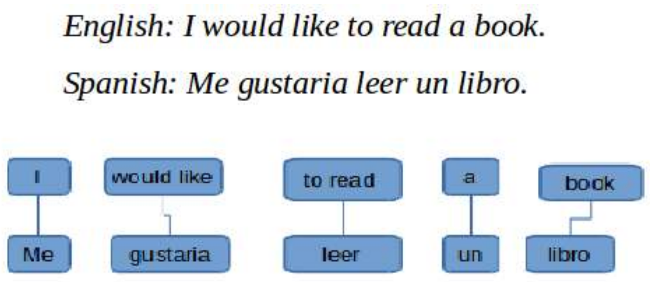

English: I would like to read a book.
Spanish: Me gustaria leer un libro.
I
would like
to read
a
book
Me
gustaria
leer
un
libro

Fig. 2.20 Tuples extraction in n-gram translation model

The probability of a sequence of tuples is then computed using a standard n-gram modelling as:

$$p(u_1 ... u_I) = \prod_{i=1}^{I} p(u_l | u_{i-1} ... u_{i-n+1}) \tag{2.9}$$

A number of additional parameters are estimated using statistics collected from a training corpus: lexical weights inside a tuple, local reorderings (also inside a tuple), distortion penalty, as well as target language model. Reordering rules are also learned from the training corpus; these rules are derived from syntactic or morpho-syntactic attributes of the training instances (Crego and Mariño, 2006) and basically represent patterns of parts-of-speech, lemmas or any other grammatical category. Decoding is performed using finite state transducers framework.

In the following listing we discuss each component of an n-gram translation model in more detail.

1. Tuple extraction.

   The first step in training an n-gram translation model is the creation of a bilingual dictionary. Almost all data driven approaches, including n-gram TM, use word alignment (automatic or manual) in order to retrieve source/target pairs. The quality of the alignment is thus of critical importance. N-gram models use a joint representation of source and target phrases called 'tuples'. During training a unique split into tuples is performed on each training instance (source/target sentence pair) and the smallest bilingual units (tuples) consistent with the alignment are retrieved (see Fig.3.14)

2. Scoring.

   The translation model is implemented as a bilingual n-gram model, i.e. a language model built on tuples. Thus the translation model probabilities at the sentence level are approximated by using n-grams of tuples:

   $$p(f_1^J, e_1^I) = \sum_{k=1}^{k} p((f,e)_k|(f,e)_{k-1}, ..., (f,e)_{k-n+1}) \tag{2.10}$$

   where e refers to a target sentence, f to a source string, $(f,e)_k$ to the k^{th} tuple of a given bilingual sentence pair, and n is the language model order.

3. Other scores.

   The consistency inside a tuple is ensured by calculating several tuple unigram scores: a source-to-target and a target-to-source lexical probabilities (IBM-1 model style):

   $$p_{IBM1}((s,t)_n) = \frac{1}{(I+1)^J} \prod_{j=1}^{J} \sum_{i=0}^{I} p(t_n^i|s_n^j) \tag{2.11}$$

   and tuple unigram maximum likelihood probabilities:

   $$\begin{array}{c} \frac{count(f,e)}{\sum_{e'} count(f,e')} \\ \frac{count(f,e)}{\sum_{f'} count(f',e)} \end{array} \tag{2.12}$$

   where f and e are source and target tuples respectively.

   These scores allow to filter out bad (inconsistent) tuples as well as tuples produced as a result of alignment errors.

The tuple bonus model (a) and target word bonus (b) model are both used in order to compensate the system preference for sentences using less number of tuples and shorter translation hypotheses.

$$\begin{aligned} &(a)\;\; p(f_1^J, e_1^I) = exp(J) \\ &(b)\;\; p(f_1^J, e_1^I) = exp(I) \end{aligned} \tag{2.13}$$

Another important set of scores is provided by the lexicalized reordering model, which predicts an orientation type of a given tuple out of four (monotone order, switch with previous phrase, forward jump, backward jump). In order to learn lexicalized reordering model, the counts of how often each extracted tuple is found with each of the four orientation types are collected. The probability distribution is then estimated based on these counts using standard maximum likelihood estimations.

4. Reordering rules.
   In n-gram MT paradigm (as of today) reordering patterns are learned from the training corpus aligned at word level and tagged with either POS tags or any other grammatical category. Thus the rules use more generic units, not the words themselves. This allows the rules to have a larger coverage. A reordering rule has the following form:

   $$t_1, ...t_n \rightarrow i_1, ...i_n$$

   where $t_1, ...t_n$ is a sequence of source POS tags and indices $i_1, ...i_n$ represent a sequence of positions into which the source words are to be reordered. Each pattern is scored with a probability computed on the basis of its relative frequency in the training corpus as follows:

   $$p(t_1, ...t_n \rightarrow i_1, ...i_n) = \frac{N(t_1, ...t_n \rightarrow i_1, ...i_n)}{N(t_1, ...t_n)}$$

5. Decoding.
   Initially n-gram models performed a monotone or a reordered search with a beam-search decoder (subsection 2.1.5). Today n-gram paradigm allows for an easy integration of FST decoding (as implemented in NCODE toolkit (Crego et al., 2011)), which in turn allows for an efficient combination of several models in the form of FST transducers. The translation process is performed as follows: first a source word graph, containing all possible reorderings (according to reordering rules) and their

respective scores, is produced; then the graph arcs are labeled with possible translation hypotheses and scores for different parameters of each hypothesis. Once the input search graph is built, it is traversed by the decoder in search of the best translation.

N-gram VS phrase-based SMT

In this subsection we will briefly outline the principal differences and similarities between n-gram and phrase-based approaches to machine translation. The major differences are:

1. in phrase-based (PHB) approach all subphrases (both short and long) consistent with word alignment are retrieved into a translation table during training, while n-gram approach (n-gram) presupposes a unique and monotonic split of an input sentence and only the shortest (more reusable) phrases are kept (see Fig.3.15). Thus n-gram tuple dictionary is much smaller than a phrase-table in PHB.

2. during decoding the same splitting scheme is applied to an input sentence in n-gram, yielding a unique and monotonic segmentation of each input sentence. During PHB decoding all possible sequences of the input sentence are considered (put on different stacks), which in turn increases the decoding time and complexity.

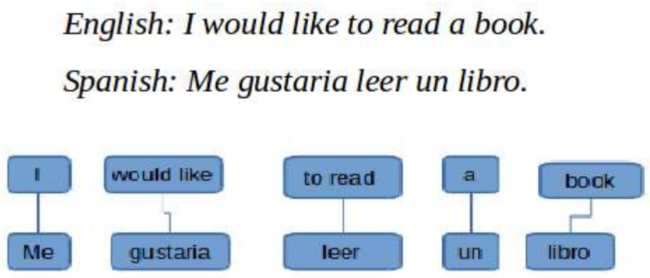
3. PHB model scoring is based on maximum likelihood estimations (relative frequency) which treat the units in isolation and do not account for the context in which they appear, while in n-gram the main translation score is an n-gram estimation allowing to take the advantage of the history (previous n tuples).

4. PHB paradigm relies on rough distance-based reordering heuristic performed during decoding. This heuristic uses simple movements which are not linguistically motivated; it assigns a reordering score based on the distance from the current position of the decoder (current phrase) thus simply punishing long distance reorderings. In n-gram paradigm reordering is performed before decoding, which drastically reduces the number of translation hypotheses to be considered during decoding. Reordering rules are patterns learned from the annotated training corpus. The patterns are based on grammatical categories (parts-of-speech, stems, semantic roles) and thus the reordering scheme has a solid linguistic foundation.

5. PHB models perform decoding using beam-search algorithm: all translation hypotheses are put on stacks and the stacks are pruned on the go in order to reduce the search space and computational complexity. N-gram models use FST decoding with fixed reordering and split.

The similar features are:

1. Lexical acquisition algorithm (alignment GIZA++). Both models use word alignments to extract bilingual translation units.
2. Log-linear core model. Both paradigms represent a log-linear combination of interpolated feature-functions; the core pipeline consists of three sub-models: phrase translation model, reordering model and language model.


English: I would like to read a book.
Spanish: Me gustaria leer un libro.
I
would like
to read
a
book
Me
gustaria
leer
un
libro

NGRAM tuples	PHB phrases
I ||| Me	I ||| Me
would like ||| gustaria	would like ||| gustaria
to read ||| leer	I would like ||| Me gustaria
a ||| un	I would like to read ||| Me gustaria leer
book ||| libro	would like to read ||| gustaria leer
	would like to read a ||| gustaria leer un
	would like to read a book ||| gustaria leer un libro
	I would like to read a book ||| Me gustaria leer un libro
	to read a ||| leer un
	to read a book ||| leer un libro
	to read ||| leer
	a ||| un
	book ||| libro
	a book ||| un libro

Fig. 2.21 Phrase extraction in n-gram VS phrase-based translation model

3. Other feature functions. Both models use a number of additional features which augment the comprehensiveness of the model with additional estimations. The scores which are common for both models are bidirectional alignment probabilities (lexical scores) and distortion penalty.

We will return to the discussion of the differences between the two translation frameworks again in section 3.2, but this time in light of NLG and their impact on the generation model performance.

2.2.4 Discriminative learning in SMT

Discriminative learning approaches have been successfully applied to many NLP tasks. This is especially appealing for machine translation, where the mapping between a source word or a phrase and its target correlate(s) seems to involve a large array of factors, such as its morphology, syntactic role, meaning, lexical context, etc.

The pioneers of discriminative learning in SMT were F.Och and H.Ney (Och and Ney, 2002). They proposed a framework based on direct maximum entropy models, and specifically a source-channel model. All additional models and knowledge sources are treated as feature functions; each feature function gets a weight according to its importance and its impact on the output quality. This approach allows a baseline machine translation system to be extended easily by adding new feature functions. The idea quickly gain popularity and a number of successful studies on discriminative SMT modeling followed, among those are (Arun and Koehn, 2007; Flanigan et al., 2013; Simianer et al., 2012) and others.

Model optimization

A number of weight optimization solutions have been proposed, among them are Minimum error rate training (MERT), MIRA (Chiang, 2012) and Pairwise ranked optimization (PRO) (Cherry and Foster, 2012). In this subsection we will take a look at one of them, MERT, which has been widely used to optimize model components' weights and which we use in our experiments.

MERT was introduced in (Och, 2003) and has since become an important component of most translation systems. As we discussed in section 2.2.1 a typical SMT pipeline represents a linear combination of feature functions which includes translation models, language models, reordering models, lexical models, etc. As a rule these feature functions are log probabilities and each of them is given a weight. The best translation is then calculated as a sum of weighted feature functions:

$$\hat{e}(f_s;\lambda_1^M) = \underset{e \in C}{\operatorname{argmax}}\{\sum_{m=1}^{M} \lambda_m h_m(e|f_s)\} \tag{2.14}$$

where $h_m(e,f)$ are feature functions and λ_m are model parameters. MERT attempts to find a set of parameters $\lambda_1,...,\lambda_n$ which will yield the best translation; it searches for weights which would minimize a given error measure, or alternatively, would maximize a given evaluation metric.

The goal is to compute the most probable sentence out of a set of candidate translations $C_s = \{e_{s,1}, ..., e_{s,K}\}$ along a line $\lambda_1^M + \gamma * d_1^M$ with parameter γ. This represents an optimization problem of the following form:

$$\hat{e}(\hat{f};\gamma) = \underset{e \in C}{\operatorname{argmin}}\{t(e,f) + \gamma * m(e,f)\} \tag{2.15}$$

where t and m are constants. Hence, every candidate translation in C corresponds to a line. Thus a piecewise linear function f' can be defined:

$$f'(\gamma; f) = \min_{e \in C}\{t(e,f) + \gamma * m(e,f)\} \tag{2.16}$$

Then Och's algorithm proceeds as follows: first $f'(\gamma; f)$ is computed for every sentence f with the incremental change in error count. This yields a sequence $\gamma_1^f < \gamma_2^f < ... < \gamma_{N_f}^f$ which denotes the interval boundaries and a corresponding sequence for the change in error count involved at the corresponding interval boundary $\Delta E_1^f < \Delta E_2^f < ... < \Delta E_{N_f}^f$. Then the optimal γ is computed by traversing these intervals while updating the error count.

Several improvements over the initial algorithm have been proposed, among them are using lattices instead of the n-best lists (Macherey et al., 2008), selecting starting points by random walks, which accelerates the convergence (Moore and Quirk, 2008) and reducing the local maximum problem through regularization and stochastic search (Cer et al., 2008).

2.2.5 Conditional Random Fields

Conditional random fields (CRFs) (Lafferty et al., 2001; Sutton and McCallum, 2006) is a statistical modelling paradigm used for structured (sequential) prediction. Notably linear chain CRF models output sequences of labels matching sequences of input instances. The 'sequential' nature of CRFs, i.e. their ability to employ context (previous or following observations), made them very popular among natural language processing researchers. CRFs showed a better performance in tasks involving sequential labelling, like named entity recognition (McCallum and Li, 2003), part-of-speech tagging (Awasthi et al., 2006), natural language understanding (Raymond and Riccardi, 2007), than previously widely used hidden Markov models (HMM). Unlike HMM, CRF models allow for more freedom in feature selection (thus defining larger feature sets) and weights assignment: the weights are not probabilities, and so do not have constrains (like $0 < \lambda < 1$). Generally, a CRF score for a sequence of labels *l* given a sequence of observations *s* is calculated as the sum of feature functions with appropriate weights:

$$score(l|s) = \sum_{j=1}^{m} \sum_{i=1}^{n} \lambda_j f_j \tag{2.17}$$

More specifically, as applied to language generation, the probability of a particular lexical representation $L = l_1, ..., l_J$ for a given semantic concept (string of concepts) $C = c_1, ..., c_J$ can be expressed as follows:

$$P(L|C) = \frac{1}{Z} \prod_{j=1}^{J} H(L_{j-1}, L_j, \phi, (c_1^N, n)) \tag{2.18}$$

with

$$H(L_{j-1}, L_j, c_{j-2}^{j+2}) = exp(\sum_{m=1}^{M} \lambda_m \cdot h_m(L_{j-1}, L_j, \phi, (c_1^N, n))) \tag{2.19}$$

where Z is the normalization factor:

$$Z = \sum_{\hat{l}_1^J} \prod_{j=1}^{N} H(\hat{L}_{j-1}, \hat{L}_j, \phi, (c_1^N, n))) \tag{2.20}$$

The weights λ for feature functions *h* are learned from the corpus through one of the optimization algorithms, like quasi-newton, gradient descend, etc.

It is computationally expensive to calculate the score for each possible sequence of labels; since there are n^m possible labels for a tag set of size *n* and a sentence of length *m*, this approach would have to check an exponential number of labels to find the one that maximizes the CRF score. For that purpose a dynamic programming algorithm is used in order to find the optimal label, similar to the Viterbi algorithm for HMMs.

CRFs for machine translation

CRFs have also been successfully applied to the task of translation (Lavergne et al., 2011). The authors recast the problem of machine translation as a sequence labelling task. They use linear CRFs (Lafferty et al., 2001) to train the main translation model and integrate it into a general n-gram pipeline, replacing the standard n-gram translation score with a CRF one. That in turn enables an easy integration of linguistic features, such as parts-of-speech, and most importantly contextual features into the translation model. The authors stress that the model estimations based on context are more reliable in case of insufficient training data and offer a richer feature set which allows the model to base the decisions on neighbouring source words.

2.2.6 MT evaluation

MT evaluation is an area of study in its own right. Evaluation quality is a subject of debate and continuous research; new automatic metrics are proposed and evaluated regularly.

Translation quality evaluation is somewhat similar to NLG evaluation: they use the same criteria and the same reference similarity metrics. Consequently they face the same problems: apparent insufficiency of automatic evaluation and the need of human expertise for reliable estimation of the output quality.

Rapid development (shorter development cycle) of modern MT systems and limitless possibilities of parameter modifications made it necessary to test MT systems regularly, quickly and at no cost. Manual evaluation is expensive to carry out after each slight modification in a model. A number of automatic evaluation metrics have been developed. Yet, like in NLG and in fact in many NLP domains, manual evaluation of the output remains the most reliable and the most informative, especially when it comes to providing feedback for error analysis. But it is also the most expensive one in terms of the time spent and required human resources. Human evaluation can either consist in assigning a score from a predefined scale (say 1-10) for different criteria, such as fluency, adequacy, content integrity, etc., to each translation or ranking of the output translations produced by different translation systems without explicitly scoring a particular sentence. Despite the virtues of human evaluation it is not feasible in continuous and rapid testing during the development cycle.

Automatic evaluation

A number of automatic evaluation metrics have been proposed; among them are some standard ones, widely used in many NLP tasks (precision, recall, f-measure) as well as metrics borrowed from a particular domain of NLP, like Word Error Rate similar to the one used in speech recognition and natural language understanding (the later calls it CER for Concept Error Rate).

- **WER**, which stands for Word Error Rate (Nießen et al., 2000), measures the number of edits required to transform a system output into one of the references. WER is calculated as follows:

$$WER = \frac{substitutions + insertions + deletions}{referencelength} \tag{2.21}$$

  WER calculation is pretty straightforward and yet it is a rather crude metric; it gives no insight into the nature of errors and it considers only one reference thus eliminating potentially correct translations which differ from that reference. Also it does not model the reordering of words and phrases in candidate translations. To address this issue another metric was proposed - **TER** - which stands for Translation Edit Rate (Snover et al.). TER allows for block movement of words, called shifts. Shifts have the same edit cost as insertions, deletions or substitutions, regardless of the number of words being shifted. Unlike WER, TER can be used with multiple references.

- TER-Plus (**TERp**) is an extension of TER that aligns words in the translation and the reference not only when they are exact matches but also when they have the same stem or are synonyms. Also it uses probabilistic phrasal substitutions in order to align phrases in the translation and the reference. These phrases are generated from possible paraphrases of the reference words. TERp uses the edit operations of TER — matches, insertions, deletions, substitutions and shifts plus three new edit operations: stem matches, synonym matches and phrase substitutions.

- **BLEU** (Papineni et al., 2002) is the most widely used (as of today) automatic evaluation metric. It represents the ratio of n-gram matches between the translation and one or several references; plus it punishes short outputs by introducing a brevity penalty (Eq.2.22).

$$BLEU = exp\left\{\sum_{n=1}^{N} w_n log(p_n) - max\left(\frac{L^*_{ref}}{L_{sys}} - 1, 0\right)\right\} \tag{2.22}$$

- **NIST** (Doddington, 2002) is a weighted version of BLEU, with more importance given to less frequent n-grams.

$$NIST = \sum_{n=1}^{N}\left\{\frac{\sum info(w_1...w_n)}{\sum_{w_1...w_n}(1)}\right\} exp\left\{\beta log^2\left[min\left(\frac{L_{ref}}{L_{sys}}, 1\right)\right]\right\} \tag{2.23}$$

NIST have been shown to have a strong correlation with human judgments, significantly higher than that of BLEU (Banerjee and Lavie, 2005).

- **METEOR** (Lavie and Denkowski, 2009) metric is calculated as the harmonic mean of unigram precision and recall, with recall weighted higher than precision:

  $$METEOR = F_{mean}(1-p)$$

  where:

  $$F_{mean} = \frac{10PR}{R+9P}$$

  P and R being precision and recall respectively, and where p is a penalty:

  $$p = 0.5\left(\frac{c}{u_m}\right)^3$$

  METEOR has also shown a higher correlation with human judgements than BLEU, specifically at the corpus level (Banerjee and Lavie, 2005).

- Recently introduced **LEPOR** metric (Han and Chao, 2012) has shown the highest correlation with human judgements. LEPOR combines sentence length penalty, n-gram position difference penalty, precision and recall:

  $$LEPOR = LP \times NposPenal \times Harmonic(\alpha R, \beta R)$$

  where LP is a length penalty which punishes both too short and too long sentences compared to the reference; $NposPenal$ is an n-gram position difference penalty designed to compare the word order in the reference and the translation, and $Harmonic(\alpha R, \beta R)$ is a weighted n-gram harmonic mean of precision and recall.

The correlation with human judgements is an important factor in automatic MT evaluation quality estimations. Traditionally the correlation is calculated using Pearson correlation coefficient (Pearson, 1895). But in many cases correlation estimations can be misleading when human judges do not show a high degree of agreement between each other.

Human evaluation

Human evaluation remains the most reliable evaluation method, despite the subjectivity and occasional divergence in judgements on the quality of a given translation.The typical criteria for human evaluation in MT are the same as for NLG:

- **Adequacy** – a judgement on how well the translation of a phrase transmits the meaning of the original, specifically it checks if the content of the source sentence has been preserved and the sense has not been distorted.

- **Fluency** is a judgement on how well-formed and fluent the translation is; it does not check the information contained in a phrase, but rather its overall grammaticalness and correctness. These, and some other criteria (like naturalness, diversity, etc.), are evaluated either by means of assigning a score within a given range to each output phrase or by ranking the phrases according to their conformity with the given criteria.

- Human-mediated Translation Edit Rate (**HTER**) is a semi-automatic metric for which humans are asked to generate a new reference translation that is as close to a given MT output as possible being at the same time fluent and preserving the meaning of the original reference. This new reference is then used as the reference translation when scoring the output using Translation Edit Rate (TER) or any other automatic metric, like BLEU. HTER is much more time consuming than TER, making it difficult to use.

In order to ensure the quality and coherence of the evaluation results, system developers calculate the inter-annotator agreement: Cohen's Kappa ((Cohen, 1960)) for two raters or Fleiss's Kappa for three and more raters. Cohen's Kappa is calculated as follows:

$$K = \frac{a_o - a_e}{1 - a_e} = 1 - \frac{1 - a_o}{1 - a_e} \tag{2.24}$$

where a_o is the relative observed agreement among raters, and a_e is the hypothetical probability of chance agreement, using the observed data to calculate the probabilities of each observer randomly saying each category. If the raters are in complete agreement then K is equal to 1. If the agreement is low (Kappa lower than 0.7), it may be the indicator of the complexity and high subjectivity of the task, but more often of an excessively broad scale of assessment scores, poorly defined evaluation criteria or lack of precision in the evaluation instructions.

2.3 Conclusion

In this chapter we presented a theoretical and historic overview of the methods that are (or were at some point) widely used in the areas of language generation and machine translation. we provided an outline of different generation and translation paradigms, as well as their core components. We also discussed a number of important issues related to language generation, such as corpora collection, learning templates/grammars from a corpus, user adaptation, output evaluation etc.

In the next chapter we will present our generation framework which uses the modeling approaches and techniques discussed here, notably n-gram translation paradigm and Conditional Random Fields - based sequential labeling models. We also resort to the commonly used evaluation metrics (BLEU) for the intermediate system assessments during the development period.

Chapter 3

Core generation model

In this chapter we present our pipeline for the task of natural language generation. As we stress throughout the thesis, we regard generation process as translation between the semantic content of a phrase and its possible lexical realizations; in our approach we adopt the techniques initially developed for statistical machine translation and also for discriminative sequence labeling. In our study we highlight the importance of understanding the specificity of generation task: how it differs from machine translation and what modifications need to be applied to the standard MT pipeline in order to make it suitable for generation. More specifically we address the issues of reordering, corpus alignment, translation/generation units extraction and selection of feature functions - all as applied to NLG. In our experiments we explore the potential of bilingual n-gram translation models and a discriminative sequence labeling paradigm, namely CRF, as well as their combination. We compare the performance of our systems with that of a traditional phrase-based SMT model which we use as a baseline. This chapter starts with a general overview of the task at hand and our proposed solutions (section 3.1). In section 3.2, we present our methodology in detail; first we review the phrase-based generation model (section 3.2.1); then we pass to the proposed alternatives: n-gram models (section 3.2.2) and the combination of the n-gram and CRF models (section 3.2.3). At each step we describe the modifications and adjustments of the default settings (designed primarily for translation) that we performed in order to make the model suit our objective. All adjustments are measured and evaluated using the BLEU score[1]: it is the subject of section 3.4 which treats the experimental part of our work. The corpora that we used in our experiments are described in section 3.3. We conclude the chapter with the discussion of results and a comparative analysis of the proposed generation models (section 3.5).

[1]We use the BLEU score only during the development stage, human evaluation of the final best-performing systems is the subject of Chapter 5.

3.1 Introduction

As we have seen in the previous chapters, there has been a growing interest in data-driven NLG models recently as they have shown their potential and their robustness in dealing with issues where rule-based generators fail (handling unseen combinations of semantic units, allowing for more flexibility and diversity in the output, etc.). We continue along this line of research and propose three novel generation pipelines: an n-gram-based generation model, a CRF-based model and the combination of the two.

Our task is to generate a coherent grammatically correct natural language sentence from a stack-based semantic representation of a dialog act:.

$$\begin{array}{l} negate(\\ \quad type = bar, \\ \quad food = russian, \\ \quad drinks = beer \\ \quad) \\ \qquad \Downarrow \end{array}$$

"There is no bar serving Russian food and beer."

Our general idea is that we regard language generation in light of two distinct paradigms: a translation between conceptual (semantic) representation of a dialog act and its lexical realization on one hand, and a sequence labelling task, where the input semantic units are tagged with the highest scoring lexical realizations according to the model, on the other hand.

Concerning our first option we selected an n-gram translation paradigm; it has a number of advantages over other SMT paradigms which can be beneficial for NLG; the most important advantage is that it explicitly models the context, specifically the history of generation - previously considered couples of source and target phrases. As our experiments have shown the n-gram translation model is very well suited for generation task and demonstrated an excellent performance on our data.

As a sequence labelling model we opted for Conditional Random Fields (CRFs). They allow us to use a rich set of features, specifically context features, and base the generation decisions on previous and following observations on both source and target sides, which was not possible with many of the previously developed data-driven NLG models (e.g. MOUNTAIN by Lagner).

Finally we combine the two models into a unique framework which allows the generator to consider the information provided by both models at generation time (along with a number of additional estimations, like target language model score) and take a more informed decision.

The techniques that we use have been previously successfully applied to different areas of natural language processing which deal with language production. For instance, n-gram model was used by (Oh and Rudnicky, 2000) to build the generation module in the Carnegie Mellon Communicator. To our knowledge, CRF models have never been applied to language generation. And yet they have been shown to have a good performance on the language understanding task (Yang and Liu, 2014), which we consider to be a reverse of NLG, and also on MT task (Lavergne et al., 2011) which is very similar to ours and is indeed our assumed general paradigm.

We propose using a modified (adjusted) pipeline of T. Lavergne and a modified n-gram translation paradigm in order to build generation models. As the models were originally developed to perform translation tasks, they are not suitable (in their classical form) for generation. We modify the default behaviour of the core models as well as their default settings in order to make the pipeline suitable for NLG.

We address a number of issues which are of major importance in language generation, particularly the difference in entropy between the conceptual (semantic) content and its lexical realization, the need for a robust long-distance reordering (at the sentence level) and a more efficient generation units extraction.

We evaluate the final combined generation pipeline as well as the base models separately (CRF and N-gram models) with a view to estimate the impact of each set of scores on the overall performance of the combined model and adjust these scores respectively. The intermediate evaluation is performed using the BLEU score which is widely used to evaluate the translation output. Obviously this score is not sufficient for the final evaluation of the best-performing combined generation systems, which should allow for multiple variants and acceptable generation outputs and also take into account various criteria other than the proximity to the reference, such as variation, naturalness, etc. Thus we perform a human evaluation of the combined CRF/N-gram system against the template-based generator and an extended version of the combined system (see chapter 4). The evaluation protocol and the results are presented in Chapter 5.

3.2 Methodology

In this section we give a theoretical ground for the generation pipelines that we propose. We start with the description of the reference model, which in our case is the classical phrase-based SMT model. It differs significantly from our proposed generation models in many aspects, including reordering scheme, generation units extraction procedure, scoring functions estimation, etc. Throughout this section we will outline the principal differences as well as similarities between the baseline and our main generation models. We will also explain why phrase-based model is not perfectly suited for our task and what problems we face when applying phrase-based SMT methodology to our data. In section 3.2.2 we present our n-gram-based generation framework. Many of the system components developed within this framework, such as bilingual language model and target language model, will be incorporated later into the joint n-gram/CRF generation pipeline. In this section we also describe our modified vocabulary extraction procedure which we further use to build the CRF-based models. Finally we examine the possibilities of combining several generation models in one common framework in section 3.2.3.

3.2.1 Phrase-based SMT model for NLG

Phrase-based machine translation has for a while been a state-of-the-art translation paradigm despite occasional criticism on the part of some researchers ((Lavergne et al., 2011; Wuebker et al., 2010)). It has also been successfully applied to language generation by (Langner et al., 2010)(see section 2.1.5). Phrase-based MT paradigm offers some interesting properties which makes it an interesting option for NLG. Among these properties are:

- building a dictionary (a table) of smaller reusable units;
- linear combination of several feature functions: translation scores, lexical consistency scores, lexicalized reordering scores, etc.;
- a basic distance-based reordering heuristics;
- integration of a target language model (TLM).

Most scores (apart from TLM and reordering scores) are calculated using standard maximum likelihood estimation formula:

$$\Phi(f|e) = count(e,f) / \sum_{i=1}^{I} f_i count(e_i, f_i) \tag{3.1}$$

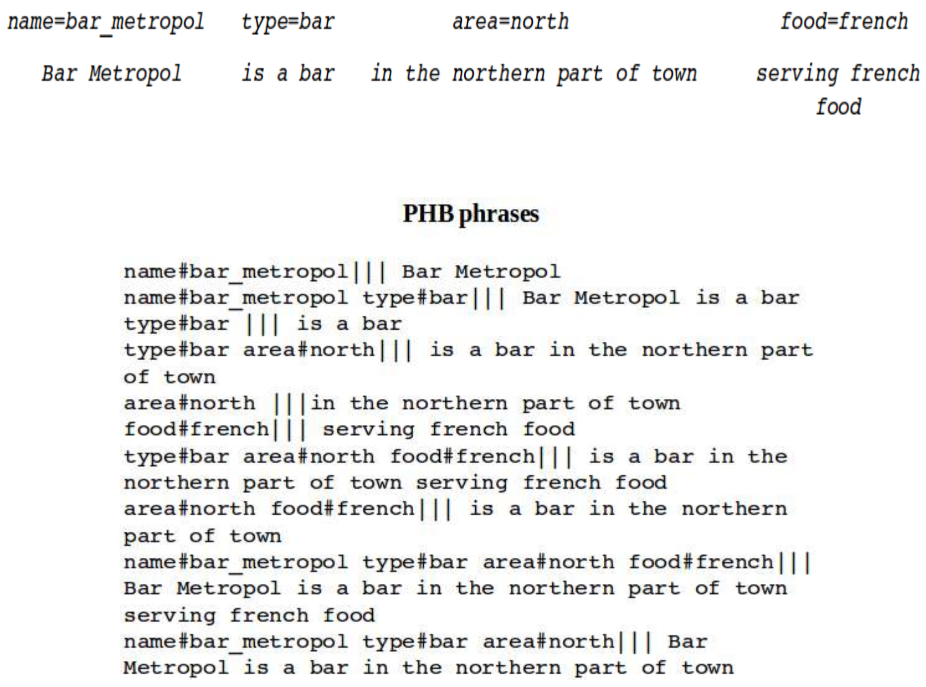
The scores are combined into an interpolated log-linear equation:

$$p(x) = exp \sum_{i=1}^{n} \lambda_i h_i(x) \tag{3.2}$$

The training process unfolds as follows:

1. first bidirectional word-to-word alignment of the training corpus is obtained (using an automatic tool, like GIZA++ (Och and Ney, 2003));

2. subphrase pairs consistent with the word alignment and all contiguous combinations of subphrases are retrieved into the translation table; if there are several concepts in the input, different combinations of adjacent concepts of different lengths are extracted into the phrase table;



PHB phrases

```
name#bar_metropol||| Bar Metropol
name#bar_metropol type#bar||| Bar Metropol is a bar
type#bar ||| is a bar
type#bar area#north||| is a bar in the northern part
of town
area#north |||in the northern part of town
food#french||| serving french food
type#bar area#north food#french||| is a bar in the
northern part of town serving french food
area#north food#french||| is a bar in the northern
part of town
name#bar_metropol type#bar area#north food#french|||
Bar Metropol is a bar in the northern part of town
serving french food
name#bar_metropol type#bar area#north||| Bar
Metropol is a bar in the northern part of town
```

Fig. 3.1 Phrases extracted during training (phrase-based model)

3. source-target subphrase pairs are assigned a translation score according to the formula 3.1 above along with several inner-phrasal lexical scores calculated in the same way;

4. an n-gram target language model is learned from the target side of the corpus;

5. reordering is based on a simple distance-based heuristic and is performed during decoding;

6. the weights for the feature functions are set with Minimum Error Training performed on a separate development set of the corpus.

A new sentence is generated from an input set of semantic concepts via beam-search decoding (see section 2.2.1). During decoding all possible splits and all possible reorderings within the input set of concepts are considered as generation hypotheses and the worst ones are pruned on-the-fly as the hypotheses stacks grow in size. We describe our phrase-based generation model and its performance on our data in section 3.4.1.

3.2.2 N-gram generation model

Unlike (Oh and Rudnicky, 2000) which used n-gram modeling in its pure form to build an NL generator, we would like to extend the n-gram generation model, adding a reordering scheme and a target LM, in accordance with a standard SMT pipeline. We follow the approach introduced in (Crego and Mariño, 2006) and build a framework which represents a combination of finite-state transducers as follows:

$$e* = bestpath(r(conc) \circ tm \circ lm) \tag{3.3}$$

where *r(conc)* is the graph containing the reordered source concepts, *tm* is the translation model based on an n-gram joint probability estimation and *lm* is a target language model. The reordering model represents the reordering rules learned from the training corpus (we describe the process in the reordering subsection below). *tm* in an n-gram model defines the joint probabilities of a given set of concepts and their lexical realization *p(conc,lex)*. The probability of a given sequence of tuples is expressed as a standard n-gram model probability:

$$p(conc, lex) = \prod_{k=1}^{K} p((conc, lex)_k | (conc, lex)_{k-1}, ..., (conc, lex)_{k-n+1}) \tag{3.4}$$

The target language model *lm* is an n-gram language model (3.5) which ensures the fluency and the overall grammatical correctness of the output (generally with n above 5, for instance 6 in our systems, see below).

$$p(lex_1 ... lex_I) = \prod_{i=1}^{I} p(lex_l | lex_{i-1} ... lex_{i-n+1}) \tag{3.5}$$

The decoding is performed on a joint Finite State Transducer (FST) graph; the role of the decoder is to find the highest scoring path in the graph which is supposed to be the best generation hypothesis according to the model. The standard n-gram translation paradigm is not well-suited for generation and needs adjustments at different levels, including the alignment, tuple extraction and scoring, etc. In the following subsections we describe every component of the framework in more detail and go over the major modifications that we applied to the standard n-gram MT paradigm to make it suitable for NLG.

Reordering

Reordering is an important step in the generation process since in our setup the semantic concepts in the input to the generator are not ordered. The way this problem is treated in a traditional phrase-based paradigm – considering all possible permutations of source-side words within a given window of words at decoding time – is not a good option as it is very expensive in terms of computational resources. Moreover the reordering decisions are taken based solely on the distance between the two units (phrases, words or concepts) under consideration. In that respect availability of the parallel training corpus does not add any information concerning reordering, despite being a rich source of linguistically motivated parallels (or patterns) between the source and the target sides.

(Crego and Mariño, 2006) proposed a reordering framework where differences in word order between language pairs are learned from the parallel corpus in the form of reordering patterns (Fig. 3.2). Patterns may be learned from any grammatical tags, like part-of-speech (POS) tags, semantic role labels, stems, etc , as well as the words themselves. Learning the patterns from more generic tags gives the patterns a more extensive coverage and a broader generalization capacity. The probability of a given pattern is estimated from pattern occurrence statistics using the maximum likelihood principle (Fig. 3.3).

Fig. 3.2 Reordering rules learned from the aligned corpus; concepts in the source side are mapped to the position indexes in the target.

price#inform type#inform name#inform -> 2 1 0
type#inform name#inform area#inform food#inform -> 1 0 2 3

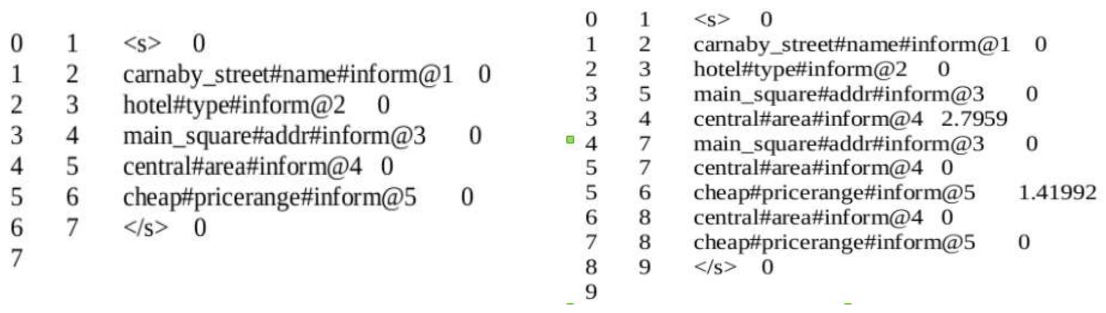
Fig. 3.3 Scoring reordering rules

$$p(t_1...t_n \rightarrow i_1...i_n) = \frac{N(t_1...t_n \rightarrow i_1...i_n)}{N(t_1...t_n)} \tag{3.6}$$

During decoding, reordering hypotheses are generated according to the extracted rules. Thus the reordering is fixed, which drastically reduces the number of generation hypotheses compared to beam-search decoding with distance-based reordering model.

For our model we use patterns of POS in the target/lexical side and reduced concepts in the source. In our scenario a reduced concept is the concept paired with the dialog act type without the associated value, eg. the concept *10#price#inform* is reduced to *price#inform*. Stripping the concepts of their values allows us to obtain more generic reusable patterns.

A monotonic search graph, produced during decoding, has a single path made of arcs with the input words in the original order. To allow for reordering, the graph is extended with reordering hypotheses, incorporating all possible reorderings of the source words established according to the reordering rules.



```
0   1   <s>   0
1   2   carnaby_street#name#inform@1   0
2   3   hotel#type#inform@2   0
3   4   main_square#addr#inform@3   0
4   5   central#area#inform@4   0
5   6   cheap#pricerange#inform@5   0
6   7   </s>   0
7
```

```
0   1   <s>   0
1   2   carnaby_street#name#inform@1   0
2   3   hotel#type#inform@2   0
3   5   main_square#addr#inform@3   0
3   4   central#area#inform@4   2.7959
4   7   main_square#addr#inform@3   0
5   7   central#area#inform@4   0
5   6   cheap#pricerange#inform@5   1.41992
6   8   central#area#inform@4   0
7   8   cheap#pricerange#inform@5   0
8   9   </s>   0
9
```

Fig. 3.4 Monotone graph VS graph with reordering hypotheses

As reordering model is integrated directly into the graph, the reordering decision is taken during decoding along with the decision on lexical choices. Hence, the best hypothesis is computed using all of system models. The monotone graph extension does not always yield good results. In some cases a large number of reordering hypotheses may create unnecessary ambiguity, which in turn would give a lot of weight to incorrect reordering patterns. One way to 'naturally' force monotone graph creation in those cases is to reduce ambiguity in the corpus used for learning reordering rules, for example by learning the reordering rules from less generic units (words/concepts themselves instead of POS or lemmas). This will result in more specific reordering rules which encode very specific and precise reordering patterns.

Alignment

Another aspect of machine translation which sometimes needs a different approach in case of generation is the corpus alignment. The quality of alignment is crucial as it defines the correct lexical content of each concept as well as the boundaries between different concepts in the lexical side of the corpus. In case of generation, it is important to make sure that the split into concepts and their respective surface forms is as fine-grained as possible, with little or no overlap between the realizations of different concepts and subsequent merges between them.

If the corpus is not aligned manually, an automatic alignment can be performed. Here again the entropy gap between the two sides of the corpus should be taken into consideration. Automatic aligners, like GIZA++ (Och and Ney, 2003), offer the possibility to choose from multiple alignment schemes, most importantly they allow to determine how permissive the produced alignment should be. As alignment is normally performed in both target-source and source-target directions, the final list of alignment points can represent either a union or an intersection of both directions. A union is a combination of all alignment points, and thus is a more permissive alignment scheme, while an intersection includes only the alignment points occurring in both lists of both alignment directions. We compare the performance of the two alignment options on our data in section 3.4.

The quality of alignment is the key factor in tuples (source/target pairs) extraction process. As a rule each concept yields several words, sometimes an entire sentence. Concepts do not always have a contiguous phrase in their lexical realization. A phrase representing a particular concept may contain words from another one (Fig. 3.5). In that case a default n-gram tuple-extraction mechanism will yield an entire set of concepts as a single tuple which is less frequent (and thus useful) as its smaller components, which in their turn will not make their way into the vocabulary.

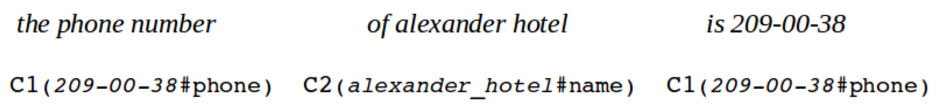


Fig. 3.5 Discontinuous lexical realization for a concept *phone* in *The phone number of alexander hotel is 209-00-38*

The issue of tuple extraction is addressed in greater detail in the next subsection.

Tuple extraction

The next step in the training process after corpus alignment is the derivation of smaller translation/generation units. (Marino et al., 2006) and (Crego et al., 2011) introduce a *tuple* as the smallest operational units. Tuple is a pair of source and target subphrases extracted in accordance with the alignment. Tuple extraction is the process of splitting aligned source-target sentence pairs into tuples. The algorithm for tuple extraction is similar to the one used in phrase-based translation models for phrase extraction and tuples are somewhat similar to the entries in a phrase-based (PHB) translation table (see section 2.2.1). Except the tuples - being the smallest units - are limited in their length and lexical coverage. Thus the major difference between the translation units extraction processes in the phrase-based model and in the n-gram model, is that in the later case only the smallest bi-units are retrieved, while in the former case all phrases consistent with the alignment, long and short, are extracted. Thus there are multiple ways to split a phrase into phrases, but only one way to split it into tuples (Fig. 3.7).

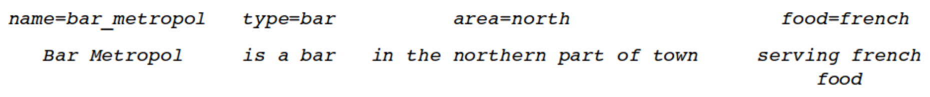
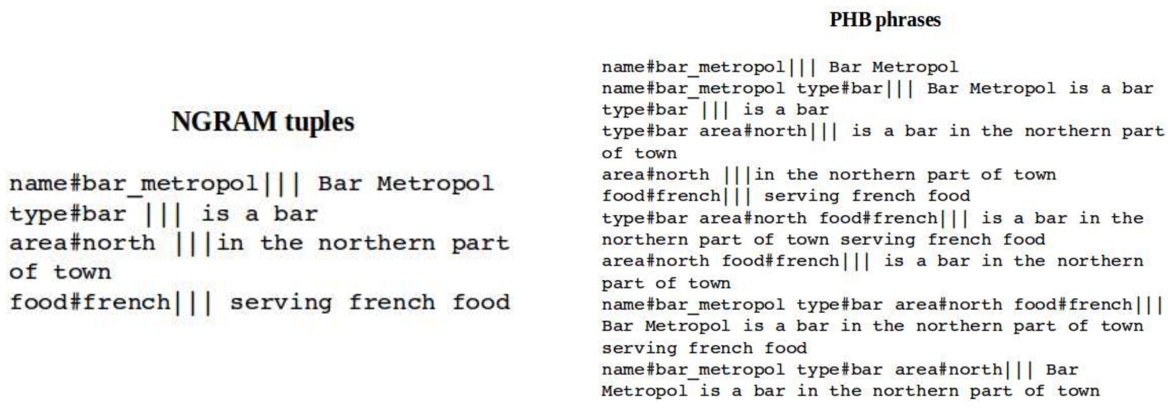


Fig. 3.6 Extraction of smallest translation pairs consistent with the alignment



NGRAM tuples

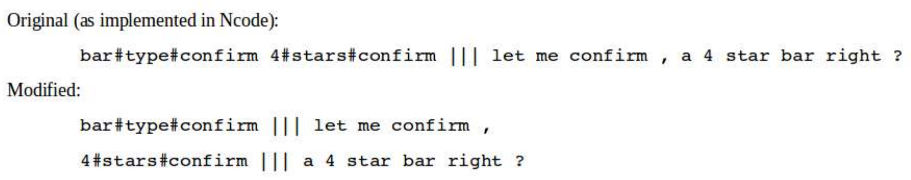
```
name#bar_metropol||| Bar Metropol
type#bar ||| is a bar
area#north |||in the northern part
of town
food#french||| serving french food
```

PHB phrases

```
name#bar_metropol||| Bar Metropol
name#bar_metropol type#bar||| Bar Metropol is a bar
type#bar ||| is a bar
type#bar area#north||| is a bar in the northern part
of town
area#north |||in the northern part of town
food#french||| serving french food
type#bar area#north food#french||| is a bar in the
northern part of town serving french food
area#north food#french||| is a bar in the northern
part of town
name#bar_metropol type#bar area#north food#french|||
Bar Metropol is a bar in the northern part of town
serving french food
name#bar_metropol type#bar area#north||| Bar
Metropol is a bar in the northern part of town
```

Fig. 3.7 Tuples VS phrases in the phrase-table

In case of generation the smallest tuple would contain one concept in the source and its lexical side (usually a group of words) in the target. We are mostly interested in retrieving such one-to-many tuples, as they are more reusable than several concepts grouped together paired with a longer phrase on the target side. Having a dictionary of such one-to-many tuples, without longer concept sequences is compensated by the translation model itself, be it n-gram or CRF model: both are based on sequential scoring, thus both make use of context (neighbouring tuples) . The original tuple extraction algorithm, as implemented in (Crego et al., 2011), does not perform the desired split into concepts as it does not assign unaligned words to one of the concepts, but instead returned the whole sequence, like in Fig. 3.8 .



Original (as implemented in Ncode):

```
bar#type#confirm 4#stars#confirm ||| let me confirm , a 4 star bar right ?
```

Modified:

```
bar#type#confirm ||| let me confirm ,
4#stars#confirm ||| a 4 star bar right ?
```

Fig. 3.8 Default VS modified tuple extraction mechanisms

One solution to this problem is the modification of the initial algorithm and forcing one-to-many tuples, i.e. tuples having one concept in the source side and a multi-word expression in the target. Such split is performed in strict accordance with the alignment regardless of inner phrasal fluency of the resulting tuples (Fig. 3.1). Smaller (the smallest in the source side) units allow for the handling of unseen combinations of concepts and are less sparse than combinations of concepts. Moreover unique (the smallest) split of the source sentence during decoding drastically reduces decoding time as there is no need to consider all possible splits of different lengths. One-to-many pairs extraction yields a dictionary where 1 concept is aligned to its multi-word lexical representation.This dictionary of extracted tuples later serves as the base for training both CRF and n-gram models.

Another problem with the default tuple extraction procedure is the so-called 'embedded concepts' or non-contiguous concepts, i.e. the concepts which contain in their lexical realizations parts of other concepts (see Table 3.1). The default behavior of the ncode tuple extractor in cases when the concepts are contiguous and each word is aligned to only one concept, is to produce the smallest units (i.e. one concept - several words) according to the underlying alignment. But this is not always the case and lexical

Table 3.1 Tuple extraction in case of embedded concepts

Input sentence:	C1 C2 C3 <=> (W1W4) (W2W3) (W5)
Default tuple extraction scheme:	C1 C2 ||| W1 W2 W3 W4 C3 ||| W5
Modified extraction scheme:	C1 ||| W1 W4 C2 ||| W2 W3 C3 ||| W5

realizations are not always contiguous. In these cases the whole block (two or more concepts and all their lexical counterparts) is extracted into a single tuple (Table 3.1). The size of the extracted tuple is very important. Long sparse tuples are in many cases unique and do not provide sufficient unit dictionary to build a robust generation model. For example: in the aligned sentence (a) concept *'phone'* is split into 2 parts: *"the phone number of"* and *"209-00-38"* (union alignment scheme).

```
(a)

Target: the phone number of alexander hotel is 209-00-38 and the price
is 10 pounds
Source: 209-00-38#phone#inform 10#price#inform alexander_hotel#name#inform
Alignment:
    NULL ({ })
    alexander_hotel#name#inform ({ 5 6 7 })
    209-00-38#phone#inform  ({ 1 2 3 4 8 })
    10#price#inform ({9 10 11 12 13 14 }) }
```

As the concept *'name'* is 'embedded' into the concept *'phone'*, in case of default tuple extraction both the concepts *'phone'* and *'name'* are extracted into a tuple creating a long instance with the lexical side of *'the phone number of alexander hotel is 209-00-38'* and the source side containing two concepts *'phone'* and *'name'*:

```
(b)
209-00-38#phone#inform alexander_hotel#name#inform ||| the phone number of
alexander hotel is 209-00-38
10#price#inform |||  and the price is 10 pounds
```

Not only the tuple is long and rare and has little chance of being seen again in the test set, but also in that case both smaller concepts which compose the long tuple are lost. The modification of the extraction algorithm into one-to-many forces ncode to split the sentence into single-concept tuples:

```
(c)
alexander_hotel#name#inform ||| alexander hotel is
209-00-38#phone#inform ||| the phone number of 209-00-38
10#price#inform ||| and the price is 10 pounds
```

It is worth noting that with the intersection alignment scheme the whole sentence is extracted into a tuple.

```
(d)
alexander_hotel#name#inform 209-00-38#phone#inform 10#price#inform |||
the phone number of alexander hotel is 209-00-38 and the price is 10
pounds
```

The concept *'209-00-38|phone|inform'* in (c) is not very grammatical but comprehensible; more importantly, in that case we also get two more valid reusable tuples that we would not obtain in case of default tuple extraction and intersection alignment scheme.

Sentences having discontiguous concepts are not numerous, yet considering (usually) a small size of the training corpus for language generation tasks, it makes sense to extract as many smaller reusable tuples as possible. Since it is impossible to reorder the lexical side of a tuple and the lexical representation of a concept is learned as a fixed sequence of words (which is a common approach shared by both phrase-based and n-gram models), there is not much to be done about the non-grammatical tuples, apart from providing inner tuple (unigram) scores and hope that they are low enough to let the model neglect these tuples during decoding.

Model estimation

The core model in the n-gram translation/generation paradigm is a Bilingual Language Model (BLM) introduced in (Marino et al., 2006). BLM represents a standard n-gram language model built on tuples – pairs of source and target subphrases or, in case of language generation, concepts and their lexical representations. The probability of a particular sequence of tuples is estimated as follows:

$$p((conc,lex)_k|(conc,lex)_{k-1},...,(conc,lex)_{k-n+1}) = \frac{count((conc,lex)_{k-(n-1)},\dots,(conc,lex)_{k-1},(conc,lex)_k)}{count((conc,lex)_{k-(n-1)},\dots,(conc,lex)_{k-1})} \tag{3.7}$$

where *conc* is the source concept and *lex* is its lexical realization. Smoothing is done using modified Kneser-Ney discounting (Kneser and Ney, 1995). The parameters are estimated using statistics collected from the parallel corpus of conceptual phrase representations (source) and their lexical realizations (target). First, a concept-to-word alignment is performed using a standard alignment procedure based on the IBM models (Brown et al., 1990) (see section 2.2.1). Then the tuples are extracted according to a standard or modified extraction scheme discussed above. The sentence-level generation model probability is calculated as a product of individual probabilities of tuples composing the sentence:

$$p(conc,lex) = \prod_{k=1}^{K} p((conc,lex)_k|(conc,lex)_{k-1},...,(conc,lex)_{k-n+1}) \tag{3.8}$$

where n is the Bilingual Language Model order, k is the index of the tuple in a given bilingual sentence pair.

Tuple inner lexical consistency

N-gram paradigm follows the maximum entropy framework (Berger et al., 1996). Apart from the main translation/generation score the system uses several additional feature functions which are combined into a weighed log-linear model (as implemented in the NCODE toolkit (Crego et al., 2011)).

As tuples represent a coupling of the source and the target sides and both are multi-word units (or, in case of one-to-many tuples, target is often a multi word expression) it makes sense to assign some additional inherent scores, like the probability of a concept having a particular lexical representation, and the reverse - a particular lexical representation denoting a particular concept. These are unigram scores, i.e. they are calculated on a tuple level and context is not taken into account. We refer to them as inner tuple scores. The probabilities are calculated on raw counts N collected from the aligned training corpus:

$$N(conc, lex) / \sum_{conc'} N(conc, lex') \tag{3.9}$$

$$N(conc, lex) / \sum_{lex} N(conc', lex) \tag{3.10}$$

In the generation scenario they represent the likelihood for a given concepts to have a particular lexical representation and vice versa the likelihood for a string of words to denote a particular concept. These scores allow to eliminate 'bad' tuples which were extracted as a result of an alignment error or inconsistent tuples obtained as a result of the enforced one-to-many extraction scheme. Also probabilities at a word level might be taken into the account. These are the equivalents of lexical translation scores in the phrase-based SMT and are calculated in the same way using IBM model-1 (Brown et al., 1990).

one-to-many

$$1/(I+1)(\prod \sum_{i=0}^{I} p_{lex}(lex_i, conc)) \tag{3.11}$$

$$1/(J+1)^I(\prod_{i=1}^{I} \sum_{j=0}^{J} p_{lex}(conc_j, lex_i)) \tag{3.12}$$

and many-to-many

$$1/(I+1)^J(\prod_{j=1}^{J} \sum_{i=0}^{I} p_{lex}(lex_i, conc_j)) \tag{3.13}$$

$$1/(J+1)^I(\prod_{i=1}^{I} \sum_{j=0}^{J} p_{lex}(conc_j, lex_i)) \tag{3.14}$$

The later scores make less sense in the generation one-to-many scenario than in the many-to-many one or in case of the translation task. We compare the impact of these scores on the model performance in both cases (generation VS translation) in section 3.4.2 below. There we also discuss how the size and diversity of the training corpus influence these scores with a subsequent effect on the overall model performance.

Another important group of scores are lexicalized reordering scores. In case of generation, lexicalized reordering probability p_o predicts an orientation type for a particular tuple. We use the NCODE implementation of lexical reordering types, with four main and two overlapped types: :

- monotone order
- switch with previous phrase,
- forward jump,
- backward jump
- discontinuous (overlapped with monotone and forward jump),
- continuous (overlapped with monotone and switch)

To estimate probability distribution p_o we count the number of times a particular tuple occurs with a given orientation type and use these counts to calculate the maximum likelihood as follows:

$$p_o(orient|conc,lex) = ((\sigma/4) + N(orient|conc,lex))/(\sigma + \sum_{o} N(o,conc,lex)) \quad (3.15)$$

where σ is a smoothing factor: $\sigma = 1/\sum_o N(o,conc,lex)$.

Decoding

Finite-state transducers provide a framework which allows to easily combine multiple models (target LM, BLM, reordering model, etc.) and perform an efficient search for the best path:

$$e* = bestpath(\Pi_2\ (perm(conc)\ \circ\ pt)\ \circ\ lm) \quad (3.16)$$

where pt is a translation score (like BLM or CRF), $perm(f)$ is a reordering score and lm is the language model score; reordering rules, learned during training are composed into a reordering transducer:

$$perm(f) = f \circ R \quad (3.17)$$

where R is estimated as follows:

$$R = O_i(r_i \cup ld) \quad (3.18)$$

Decoding process is performed on already segmented and reordered source sentences. Thus we do not need to handle all possible segmentation variants and all possible permutations in the source sentence which makes decoding computationally expensive (like in a phrase-based approach). The information about longer sequences that is lost due to the predefined split is compensated by calculating tuple n-gram model scores.

The decoding process unfolds as follows: first a test source sentence is transformed into a word graph combining all possible reorderings (according to the reordering rules learned from the training corpus) along with associated reordering scores. This graph is then monotonously decoded. The final generation hypothesis score is composed of several principle model scores: 4 tuple unigram scores, 6 lexicalized reordering scores, the main generation model (n-gram or CRF) and the target language model. During decoding a number of additional scores may also be employed : a tuple bonus model (a) and target word bonus model (b) are both used in order to compensate the system preference for sentences using less number of tuples and shorter translation hypotheses.

$$p(conc_1^J, e_1^I) = exp(I) \tag{3.19}$$

$$p(f_1^J, e_1^I) = exp(J) \tag{3.20}$$

The word bonus model is a counterbalance to the target language model which gives higher scores to shorter sentences.

A target-language model is estimated as an n-gram language model over the target words.

Finally the joint graph is searched for the best path according to the combination of all models and scores described above.

Tuning the weights

As the generation/translation pipeline represents a linear combination of several feature functions, the Minimum Error Rate Training (MERT) (Och, 2003) algorithm may be used in order to learn the optimal weight for each function which would maximize the BLEU score on the development set.

3.2.3 Combination of CRF and n-gram generation models

The discriminative version of *tm* in the initial combination of transducers 3.21 models the conditional probability of the lexical side given a set of concepts, i.e. *p(lex|conc)*. The conditional probability of the lexical realization given a concept can be modeled with

Conditional Random Fields. Thus the discriminative pipeline is obtained by replacing the original n-gram model with the CRF-based one.

$$e* = bestpath(r(conc) \circ tm \circ lm) \tag{3.21}$$

Given matched sequences of observations x^L and labels y^L CRFs expresses the conditional probability of labels as:

$$P(x_1^L|y_1^L) = \frac{1}{Z(x_1^L;\theta)} exp(\theta_T G(x_1^L, y_1^L)) \tag{3.22}$$

The joint version of the pipeline, which integrates both n-gram and CRFs, represents a combination of transducers which can be expressed as follows:

$$e* = bestpath(r(conc) \circ tm_1 \circ tm_2 \circ lm) \tag{3.23}$$

where tm_1 is an n-gram model graph and tm_2 is a graph containing CRF scores. The impact of each model and a corresponding weight is learned through Minimum Error Rate training performed on a development set.

3.3 Corpus

In this section we describe the corpus that we used in our experiments, as well as the underlying semantic formalism.

3.3.1 Original data

To train and test our NLG models we opt for the TownInfo corpus (Boersma and Weenink, 2008), both English and French versions. This corpus was created for the CLASSiC project; it represents a collection of tourist information dialogues, which imitate the interaction between a user and a dialog system. The utterances in these dialogs are mapped into their semantic representation - dialog acts. Each dialog act contains a frame and a number of semantic concepts in the form of key=value pairs, e.g.:

> Dialog act: $inform(name = bar\_metropol, type = bar, area = north, food = french)$
> Utterance: *Bar Metropol is a bar in the northern part of town serving french food.*

The in-house generation module uses a set of templates in order to generate a response during system-user interaction. Templates represent a coupling of a given semantic content (frame along with a set of concepts) and a corresponding lexical form (sentence). As we are mainly interested in the system responses, we use these templates along with the database of concept values to build the training instances by replacing the slots in the templates with the corresponding values. We also used the filtered speaker generated utterances from the TownInfo corpus and their semantic counterparts in order to augment the system vocabulary (the training corpus). The French version of the corpus was created from templates manually translated into French. The French corpus was build in the same way, except it does not contain user-generated utterances. Working with two different languages allows us to estimate the potential of our approach as a 'language-independent' paradigm.

3.3.2 Semantic formalism

In the stack-based semantic formalism, used in the TownInfo corpus, each dialog act is represented with a dialog act type (a pragmatic frame) and a stack of semantic concepts in the form of key-value pairs.

Frames

inform()	provide information (affirmative statement)
select()	offer a selection ($A or $B)
reqmore()	demand additional information
...	

Concepts

type	*hotel, bar, restaurant, ...*
name	*Botcka, Alexander, Chez Sergu, ...*
price	*chip, expensive, moderate, ...*
...	

There are 16 main dialog act types in the system side of the corpus:*inform, select, request, negate, confirm, affirm, hello, bye, repeat, thankyou, restart, help, reqalts , reqmore, ack, deny*. The list of concepts includes 23 units, e.g. name (like in «Chez Sergu is a bar»), price (like in «the prices are cheap»), area (like in «in the eastern part of town»), etc. This formalism was developed within the framework of the CLASSiC project dialogue act scheming.

3.3.3 Building TownInfo-based corpora

The initial NLG module in the in-house dialog system was template-based: it used a set of predefined templates for specific concept combinations. Not all possible combinations have been covered and the templates did not change apart from different values in variable slots. Our goal is to replace template-based system with a data driven one; but as we already have some templates at hand we used them as a basis for building a small training corpus, that we envision to extend later on.

The initial format on the source side is a stack of concept=value pairs combined within a parent concept framing the dialog act and denoting the discourse type of the generated sentence. This format is not suitable for our method: we want to have the possibility to align each concept-value pair to its respective lexical realization at the same time preserving the information about the discourse type of the entire stack (semantic frame). Thus we convert the stacks into a factored representation as follows:

Initial format: $confirm(type = restaurant, food = french, drinks = dontcare)$
Factored format: *restaurant#type#confirm french#food#confirm dontcare#drinks#confirm*

This format helps us keep all necessary information and at the same time allowing for an easy handling (dropping/adding/replacing) of factors. Specifically, this format permitted us to drop the values from concepts when building a pattern-based reordering model, which drastically reduced the number of patterns and yielded more generic and permissive reordering graphs[2]. This format also allows us to drop the first two factors and keep just the value in the output in case of unknown factor combination (OOV concepts).

Initially we had at our disposal the following datasets:

1. a list of templates, 121 in total,in the following form:

   $confirm(type = \$U, food = \$W, drinks = dontcare) \rightarrow$ "Let me confirm, you are looking for a $U serving $W food and any kind of drinks right?"

2. a database of values
   For each concept we had a set of possible values. Having the values separated from the templates is an advantage as the value set can be treated (extended/filtered) separately.

   $area = ("central"|"east"|"north"|"riverside"|"south"|"west")$
   $pricerange = ("cheap"|"expensive"|"moderate")$

[2]As our experiments have shown later on, this features was detrimental to the overall performance of the system.

3. a corpus of utterances produced by users
   Besides templates and concept values we had a corpus of user-generated utterances labelled with respective concept tags.

   $affirm(type = hotel, pricerange = cheap, stars = 2) \rightarrow$ "ok i'm looking for a basic two star hotel";

   This corpus is normally used for training language understanding modules. It cannot be used 'as is' for training the generation systems: syntactic structures and most lexical units are not typical of the system. At the same time it is an invaluable source of new concept values (like names, addresses, etc). Ideally the training data should be augmented with variable values of that sort, and not with the structures, inherent for users. Thus filtering user utterances and retrieving neuter values which are identical for both the system and the user, might be beneficial for our corpus. We used user-generated utterances to augment the set of values in our database. The filtering is performed as follows: first we convert the semantic stacks into the factored format; then we run GIZA++ to obtain the alignment between the concepts and their lexical realizations; after that we extract all the concept-string pairs consistent with the alignment, which do not contain first person pronouns («I», «me» «my»)[3].

We built the main corpus as follows: for each concept in a given template we inserted one-by-one the possible values into the slots, thus obtaining different combinations of values. For example:

Initial template:
$inform(type = \$U, name = \$X, pricerange = \$Y) \rightarrow$ '\$X is a \$U in \$Y pricerange.'
Replaced values:
$inform(type = \$U, name = \$X, pricerange = cheap) \rightarrow$ '\$X is a \$U in cheap pricerange.'
$inform(type = \$U, name = \$X, pricerange = moderate) \rightarrow$ '\$X is a \$U in moderate pricerange.'
$inform(type = \$U, name = \$X, pricerange = expensive) \rightarrow$ '\$X is a \$U in expensive pricerange.

We do not really care about the validity of the combinations of values, for example if *Chez Sergu* is really a bar in a moderate price range. The important thing is to obtain small further reusable units, like *is a bar, in a moderate price range*, etc.

[3]this method works fine on our data (the English part of TownInfo) because there is no variation in verbal forms between the first and the second person, both singular and plural, in English. It will not work on any other language where the distinction between verbal forms are manifest.

Fig. 3.9 Factored concept representation: first factor is a value (*restaurant,north*); second is a concept (*type,area*) and the third is a discourse frame (*inform*)

Source: *restaurant#type#inform north#area#inform*
Target: *a restaurant in the north part of town*

We used this procedure to build the training data for both target language model and generation model. We ended up with 2476 training instances in the original corpus (without user data). Then we added utterances produced from the user data via filtering (1151). We split the final corpus into a training set (3151 sentences), a development set and a test set (230 and 200 sentences respectively). In the next chapter we describe the methodology that we used in order to extend and diversify this initial corpus by adding new lexical units and new syntactic constructions.
There are a number of advantages in having the training corpus built this way. As opposed to user-generated, user-collected, crowd-sourced data, our corpus does not need extensive validation; the proper structures of templates ensure grammatically correct sentences; moreover, there is no noise, no user-specific interjections (as in case of registering user-produced utterances), etc. The main disadvantage of this methodology is that it requires an initial hand-crafted set of templates. But unlike producing templates for template-based generator, in our case there is no need for an exhaustive set of templates: the only requirement is that it contains all the concepts and all the frames available to the system , not necessarily all possible combinations of concepts or all possible values for each concept. A thoroughly thought out template set may contain no more than 30 templates (see 3.10). It may contain only one example realization for each concept, which is extracted during the training and reused in different combinations later on. This approach requires only a small basic set of templates and a database of values which is extended further automatically. We believe that this method for corpus building is ideal for a limited domain generation system. In our actual template set most templates are redundant, i.e. the same keys (concepts) having the same surface realization occur in different combinations. When building a data-driven model it is enough to have one template containing all the concepts within a given context instead of a number of smaller templates with repeating concept combinations.

When fed to the tuple extraction program, the later template will produce exactly the same number of distinct tuples as the above smaller templates. Different realizations for different tuples extracted from this template can be later obtained using automatic paraphrasing (see section 4.3.2).

Fig. 3.10 Template recombination.

Initial set:

$inform(type = \$U, name = \$X) \rightarrow$ '\$X is a \$U.'
$inform(type = \$U, name = \$X, food = \$W) \rightarrow$ '\$X is a \$U which serves \$W food.'
$inform(type = \$U, name = \$X, pricerange = \$Y) \rightarrow$ '\$X is a \$U in \$Y pricerange.'
...
Recombined (joined) templates:

$inform(type = \$U, name = \$X, area = \$V, food = \$W, pricerange = \$Y) \rightarrow$ '\$X is a \$U in \$V part of town in \$Y pricerange which serves \$W food.'

3.3.4 Subconcept format

In order to improve the alignment and also for more precise lexical selection we adopt another form of semantic representation: we generate a subconcept for each potential word in the target side. As we do not know in advance the number of subconcepts that a concept might produce, we train a fertility model which generates subconcepts from concepts based on the frequency counts (see below). Our experiments show that this is a promising direction, though it largely depends on how precise subconcept generation model is.

Fig. 3.11 Subconcepts

Ref: the phone number of metropol is 378-00-89
Sem: metropol#name#inform 378-00-89#phone#inform

Subconcepts:

(the)	(phone)	(number)
378-00-89#phone#inform	378-00-89#phone#inform	378-00-89#phone#inform
378-00-89#phone#inform	metropol#name#inform	378-00-89#phone#inform
(of)	(metropol)	(is)
378-00-89#phone#inform		
(378-00-89)		

On the other hand it leaves no space for ambiguity and, more importantly variability, when each concept has a specific number of words in its surface realization. And variability and flexibility are among the main reasons to adopt data-driven paradigm rather than rule-based approach in the first place. Nevertheless we do perform a number of experiments where we generate and integrate subconcepts in order to estimate their potential.

On the other hand the idea of having an intermediate generation step which produces the number of subconcepts in the source concept stack equal to the number of words in the target seems very attractive, as it makes the generation process more flexible: instead of memorizing the entire blocks of words for each concept, like *«near the Main Street»*, we could have smaller and thus more reusable units, which occur more frequently. That would be especially useful in case of non-contiguous concepts, like *378-00-89#phone#inform* in Fig. 3.11 .

Unfortunately it is difficult to predict the exact number of output words in each case and adding several possibilities increases the complexity of the model. Nevertheless we take a closer look at subconcept generation; we develop a very simple generation algorithm in order to estimate the potential of subconcept integration. We elaborate on our basic fertility model in the rest of this section.

Fertility model: subconcept generation

As we said earlier, our fertility model is very simple (for the purpose of quick implementation and testing). We record all concepts in the training set along with the number of words they generate. Thus we obtain a sort of fertility table which can be used to generate subconcepts in the test set. In its simplest form the number of subconcepts with the highest occurrence is selected. As a rule there is little ambiguity in the length of the produced string, as most concepts generate a certain number of words (at least in our small, rather monotone corpus), cf:

restaurant#type#inform ||| a restaurant	(2 words)
restaurant#type#inform ||| any restaurant	(2 words)
north#area#inform ||| in the north part of town	(6 words)
north#area#inform ||| near the north part of town	(6 words)

We evaluate our simple fertility model by comparing the number of subconcepts generated by the model and the number of subconcepts in the reference. Thus we end up with two formats of the training corpus. In the following sections we refer to the corpus containing subconcepts on the source side as TownInfo English (subconcepts) and TownInfo French (subconcepts).

Subconcept Error Rate (SER)

We calculated SER on the testset of 230 sentences, containing on average 2.9 concepts each (the number of concepts per sentence varied from 1 to 6); thus for each concept in each sentence we generated subconcepts (or 'lexicalized concepts'), according to our model. We ended up with a total number of 3001 subconcepts. We compared the total number of subconcepts with the number of words in the reference, which equals 3028. We counted 289 deletions, 316 insertions and 0 substitutions and got the SER=0.201. As we can see, just by picking the most frequent number of words generated by a given concept, we are able to achieve a rather high accuracy of our fertility model. We test the performance

Table 3.2 Subconcept Error Rate (TownInfo English)

SER	**0.201**
Delitions	289
Substitutions	0
Insertions	316
Total words	3028

of this format of the corpus on the CRF generation model outside of our main pipeline and on ncode generation model. We compared the performance of the models trained on the corpus produced using our subconcept generation method with the performance of the standard corpus as well as with the performance of the corpus built using the number of subconcepts in the reference. The results are presented in section 3.4. Despite a somewhat worse performance on our data, this direction is quite promising and certainly offers more flexibility and variability on the lexical level within the concepts as it splits the concept into smaller more reusable units, thus allowing for unseen lexical combinations instead of just manipulating fixed multi-word units representing a concept.

3.4 Experiments and results

In this section we will present the experimental setup for each group of NLG models discussed in section 3.2.

3.4.1 Phrase-based generation model

We start off by building a phrase-based generation model for each corpus using the MOSES toolkit (Koehn et al., 2007), which will serve us as reference models. We keep all the default settings, including the intersection alignment switch[4].

The results are summed up in Table 3.3. As we can see even a simple phrase-based model built on TownInfo (English) corpus using standard settings yields the BLEU score of 0.67. This indicates that even with a simple phrase-based system we are able to gain the performance comparable to that of the template-based model. By adding user data we increase the BLEU score by 1 point (0.68) on the same data. The performance of phrase-based models on the French version of the corpus is significantly worse than on its English counterpart. This can be explained by a notable difference in vocabulary size and in the use of certain grammatical categories, like gender and number (e.g. presence of gender and number markers in articles, adjectives and some numerals).

Table 3.3 BLEU scores for phrase-based baselines

Corpus	**BLEU**
TownInfo (English)	67.2
TownInfo (English) + spk	68.2
TownInfo (French)	51.9

In general, phrase-based baselines show a rather decent performance; they are easy to construct, flexible and robust when it comes to treating unseen combinations of concepts, contrary to rule-based models. The negative points include poor handling of reordering and an overall lack of control over the decoding process (much less than in n-gram MT). Out-of-context maximum likelihood scoring is also debatable as for its reliability and adequate model estimations.

3.4.2 N-gram generation model

In this section we present our experiments with bilingual n-gram (BLM) models, which constitute the first of the two core models in our generation pipeline. Specifically, we discuss issues like alignment, reprdering and tuple extraction and their impact on the model performance (as applied to our data). We compare the performance of the n-gram model which uses the default tuple extraction scheme against the ’one-to-many’ model and the

[4]In phrase-based models, as they do not involve tuple extraction, the intersection alignment scheme shows the best performance

performance of both tuple extractors (default and modified) on the corpus aligned with a union versus an intersection alignment schemes. We also study the effect of the n-gram length on the model performance, comparing tuple unigrams with bigrams and trigrams. Another issue that we closely examine is the impact of the lexical (tuple unigram) scores on the model performance, and how these scores change with the increase of the corpus size. We end this section with a comparative analysis of BLM models in translation versus generation tasks.

Alignment

In our experiments we test two alignments schemes: intersection between the source-target and target-source alignments and their union. Union alignment scheme is known to have a higher recall than the intersection scheme; but it also shows lower precision. In our case the recall is more important as it provides additional alignment points which might be used to resolve the problem of embedded concepts.
For example in intersection alignment output in Fig. 3.12, the first word of the sentence *they* is aligned to the concept *chez_sergu#name#inform*. This is the correct alignment point, nevertheless this creates an embedded concept *folk#music#inform* inside the concept *chez_sergu#name#inform*. Thus both are retrieved into the tuple along with their lexical counterparts. If instead of intersection we take the union (Fig. 3.13) of alignment directions the additional alignment point for *'they : folk#music#inform'* is created, thus enabling our tuple extraction mechanism to split the couple of concepts into two different smaller tuples. In general, union alignment produces more one-to-many smaller tuples and, therefore, less unique composite tuples which are less reusable.

Fig. 3.12 Tuples produced with intersection alignment scheme

Reference: *they play good folk music at chez sergu and it is a family style hotel*

Concepts: chez_sergu#name#inform folk#music#inform is_a_family_style_hotel#comment#inform

Tuples extracted from the alignment:

chez_sergu#name#inform folk#music#inform ||| *they play good folk music at chez sergu*

is_a_family_style_hotel#comment#inform ||| *and it is a family style hotel*

Reference: *sorry there is no bar in the moderate price range in the west part of town*

Concepts: none#name#inform bar#type#inform west#area#inform moderate#pricerange#inform

Tuples extracted from the alignment:

none#name#inform ||| *sorry there is no*

bar#type#inform ||| *bar*

west#area#inform moderate#pricerange#inform ||| *in the moderate price range in the west part of town*

Fig. 3.13 Tuples produced with union alignment scheme

Reference: *they play good folk music at chez sergu and it is a family style hotel*

Concepts: chez_sergu#name#inform folk#music#inform is_a_family_style_hotel#comment#inform

Tuples extracted from the alignment:

folk#music#inform ||| *they play good folk music at*

chez_sergu#name#inform ||| *chez sergu*

is_a_family_style_hotel#comment#inform ||| *and it is a family style hotel*

Reference: *sorry there is no bar in the moderate price range in the west part of town*

Concepts: none#name#inform bar#type#inform west#area#inform moderate#pricerange#inform

Tuples extracted from the alignment:

none#name#inform ||| *sorry there is no*

bar#type#inform ||| *bar*

moderate#pricerange#inform ||| *in the moderate price range*

west#area#inform ||| *in the west part of town*

What is notable, is that just by modifying the alignment scheme we can achieve a significantly better performance (see Table 3.5), even with the inherent n-gram tuple extraction algorithm. We will return to the subject of tuple size and tuple extraction in the following subsection.

Table 3.4 Number of tuples extracted as a result of different alignment paradigms

alignment	**total N tuples**	**unique tuples**
intersection	7201	635
union	7422	597

Tuple extraction procedure modifications

We compare the performance of the n-gram model which uses tuples extracted with a default n-gram extraction procedure as implemented in NCODE toolkit with a modified 'one2many' extraction mechanism. Default extraction is tested on the original corpus (simple concepts) as well as on the lexicalized version (with subconcepts). We also study the impact of different alignment schemes: union of source-target and target-source alignment points which is more permissive and their intersection which is more restrictive. The results are presented in Table 3.5.

Among the tuples extracted using the original algorithm there were a significant number of multi-concept tuples, which contain several concepts on the source side, even when using the union alignment. In several cases, entire sentences were extracted into single tuples. Such

Table 3.5 Original VS modified tuple extraction, intersection VS union alignment schemes (BLEU scores, TownInfo(English). Complete n-gram framework (all scores)

Model	**intersection**	**union**
subconcept tuples	24.5	67.9
tuples original	18.1	37.3
tuples one2many	56.1	68.8

tuples have a very slim chance to be used during decoding as their source side represents a specific combination of concepts and values which is unlikely to occur in the test set. The union alignment produces less complex multi-concept tuples which boosts the performance of the standard model to 37.3 in BLEU. Subconcept model performance is significantly better due to a greater number of smaller reusable units and a greater number of tuples (per phrase) with a higher chance of occurrence in the test set. The modified one2many extraction model achieves a significantly higher BLEU score even with the intersection alignment (compared to the other two models) and with the union alignment it outperforms the model based on subconcepts by 1 point.

Tuple bigrams VS trigrams

Another model parameter that we want to examine closely is the effect of the n-gram model order on the overall model performance. Thus unigram model amounts to using isolated tuples and n-gram model would employ tuple combinations of maximum length of n, with possible interpolation of several orders (n). The results of different language model orders are summed up in Table 3.6.

As we can see, unigram model yields 31.33 in BLEU. Interestingly there is little difference in the performance of the 3-gram and 2-gram models on our data.

Table 3.6 The effect of the n-gram order on the model performance, BLM model alone, TownInfo (English)

BLM order	**BLEU**
unigrams	31.3
2-gram	82.0
3-gram	84.4

Target LM

Another important component of our generation pipeline is the target language model (TLM). We build a 6-gram TLM from the target side of our training corpus (3151 sentences). In order to assess the impact of TLM on the performance of the generation pipeline we carry out a number of experiments coupling TLM with BLM of different orders. The results are presented in Table 3.7.

Table 3.7 The impact of the target language model on the performance, BLM + TLM, TownInfo (English)

BLM order	TLM	BLEU
unigrams	yes	56.6
2-gram	yes	81.3
3-gram	yes	82.7
no	yes	75.7

Another interesting point is that language model degrades slightly the performance of the n-gram model in case of 3-grams and 2-grams, but its impact is extremely small. On the other hand it improves significantly the performance in case when BLM model is removed and the decoder has to fall back to tuple unigrams. Also it is remarkable, that using target language model alone yields 75.7 in BLUE. It means that simple 6-gram modeling of the target side is sufficiently representative of the data that we work with.

Reordering

Reordering model is one of the core components of our generation pipeline (section 3.2.2). It represents a graph containing the reordering hypotheses of the input concepts. The effect of the reordering model largely depends on the training corpus format (source/target units representation). Generally, more specific source corpus representation yields less ambiguous reordering model and vice versa, generic tags in the source (like POS, stems, etc) produce a more complex reordering model which might be good in the scenarios, where the source side requires extensive reordering, but on our rather monotonous data, diminishing the impact of the reordering model, or eventually performing monotone decoding, yielded the best results. One way to 'naturally' force monotone graph creation is to reduce ambiguity in the corpus used for learning reordering rules. We replace more generic reduced concepts with their full counterparts, i.e. concepts with values. Using full-blown concepts with values on the source side along with POS on the target side reduces ambiguity and consequently diminishes the number of reordering variants. Thus in our case learning the model on the tokens themselves (concepts on the source side) resulted in a monotone reordering model which showed a better performance on our data.

Table 3.8 The impact of the reordering model trained on different source formats, BLM alone, TownInfo (English)

BLM order	**BLEU**	
	monotone R	**reducedConc R**
unigrams	31.3	30.9
2-gram	82.0	69.2
3-gram	84.4	70.7

In our case a large number of reordering patterns learned as a result of using ’reduced’ concepts on the source side of the training corpus created unnecessary ambiguity, which in turn gave a lot of weight to incorrect reordering hypotheses. By eliminating reordering variants we were able to boost the BLEU score by points. We came to the conclusion that a ’loose’, permissive reordering model worsens the performance of the model on our data and that monotone search is preferable.

Lexical scores

The last component of the n-gram pipeline, or rather a set of components, is the tuple unigram scores (lexical translation and lexical reordering scores). As with previous model components, we want to estimate the influence of these scores on the overall model performance. In this case we compare the results of the model trained on the English corpus with the model trained on the French part of the corpus. We carry out a number of experiments where we manually set the weights for these scores within the log-linear combination of all model components (bilingual language model, reordering model, target language model and lexical scores) and examine their effect on the performance.

Unigram tuple scores and lexical reordering scores, when used with their full capacity (the weight of 1.0) degrade significantly the performance of the system for both the English and the French versions of the TownInfo corpus. There is a significant 20-point difference between the models using tuple unigram (lexical) scores and the models where the weights for these scores are lowered to 0.001. This clearly demonstrates a negative impact of lexical scores in the current setup (the original, non-extended training corpus). The nature of our training corpus, specifically a very low level of variance in the vocabulary, results in ’swollen’ lexical scores which in turn affects the performance of the entire pipeline. We discuss this issue in more detail in the following subsection. It is worth noting that for the French part of

Table 3.9 N-gram model for generation task: different parameter combinations (weights) (TownInfo English)

tuple uni.scores	**lex.reord. scores**	**BLM**	**LM**	**BLEU**
1.0	**1.0**	**1.0**	**1.0**	68.8
0.001	**0.001**	**1.0**	**1.0**	84.5
0.0	0.0	**1.0**	0.0	82.0
0.0	0.0	0.0	**1.0**	75.7
1.0	0.0	0.0	0.0	31.3
0.0	0.0	**1.0**	**1.0**	82.7
1.0	**1.0**	0.0	0.0	46.4

the corpus the difference is not that significant as for the English part. This can be explained by the difference in the vocabulary size, and subsequently, more balanced lexical scores. This highlights once more the importance of training corpus extension and diversification, that we discuss in the next chapter.

Table 3.10 N-gram model for generation task: different parameter combinations (TownInfo French)

tuple uni.scores	**lex.reord. scores**	**Bilingual LM**	**LM**	**BLEU**
1.0	**1.0**	**1.0**	**1.0**	59.7
0.001	**0.001**	**1.0**	**1.0**	67.4
0.0	0.0	**1.0**	0.0	56.8
0.0	0.0	0.0	**1.0**	69.1
1.0	0.0	0.0	0.0	44.1
0.0	0.0	**1.0**	**1.0**	63.4
1.0	**1.0**	0.0	0.0	44.8

Combining all model components

Finally we combine all features into the complete pipeline; we perform Minimum Error Rate training in order to find the optimal weight for each feature function. Our final model is built on tuples obtained with the 'one2many' extraction procedure based on the union alignment scheme. We opted for a monotone reordering as the optimal reordering solution. The results are presented in Tables 3.11 and 3.12.

Table 3.11 N-gram model for generation task: different parameter combinations (MERT) (TownInfo English)

tuple uni.scores	**lex.reord. scores**	**Bilingual LM**	**LM**	**BLEU**
1.0	**1.0**	**1.0**	**1.0**	68.8
0.001	**0.001**	**1.0**	**1.0**	84.5
MERT	**MERT**	**MERT**	**MERT**	69.2

Table 3.12 N-gram model for generation task: complete pipeline (TownInfo French)

tuple uni.scores	**lex.reord. scores**	**Bilingual LM**	**LM**	**BLEU**
1.0	**1.0**	**1.0**	**1.0**	59.7
0.001	**0.001**	**1.0**	**1.0**	67.4
MERT	**MERT**	**MERT**	**MERT**	61.2

Translation VS generation: n-gram model

We want to compare the performance of the n-gram model applied to language generation as opposed to the translation task.

We replicated our experiments on a subset of French-English News-Commentary corpus from WMT'2011 data set which has the size of 115K sentences. This subsidiary study will allow us to estimate the importance of different feature functions in both scenarios: translation and generation. In our experiments the weights for different feature functions are equally set to either 1.0 or 0.0, which amounts to using or not using a specific feature (score) The results are presented in Table 3.13.

Interestingly unigram tuple scores seem to be more informative and reliable than target LM or even bilingual 3-gram LM for the translation task in our setup. One possible explanation for this is the small size and the consequent sparsity of the training corpus: tuple unigrams are small and occur more often than their 2-gram and 3-gram combinations; thus inner tuple scores have at their disposal a richer statistics for probability calculation than 3-gram BLM. Another explanation concerns the scores themselves and the way they are obtained: considering a large selection of target choices for a particular source unit (large number of tuples) in case of translation, lexical probability distributions among the tuples are well-balanced.

In case of generation the dynamics are different. Here lexical scores add very little information to the model, and even hinder the performance of BLM, which shows better results in isolation from lexical unigram scores. The explanation for that is again the quality of the training data. Generation tuples do not show the same level of variability and tend

Table 3.13 The impact of different feature functions on the French-English n-gram translation model

tuple uni.scores	**lex.reord.scores**	**Bilingual LM**	**LM**	**BLEU**
1.0	**1.0**	**1.0**	**1.0**	13.7
0.0	0.0	**1.0**	**1.0**	2.1
0.0	0.0	**1.0**	0.0	2.4
1.0	0.0	0.0	**1.0**	13.1
1.0	0.0	0.0	0.0	10.3
0.0	0.0	0.0	**1.0**	1.6
0.0	**1.0**	0.0	**1.0**	1.5
0.0	**1.0**	**1.0**	**1.0**	2.0
1.0	0.0	**1.0**	**1.0**	14.2
1.0	**1.0**	0.0	0.0	11.0

to repeat often in the same (or similar) tuple combinations. As there are few tuple variants BLM was able to gather sufficient statistics for most of them. At the same time tuple lexical models yield unbalanced estimations.

To demonstrate this we selected two example tuples, one from the translation and another one from the generation tuple dictionary. A very frequent concept *'bar#type#inform'* is represented in just 6 different tuple variants while a rather infrequent word *'classe'* has 26 different translations.

Table 3.14 Tuples for the concept *'bar#type#inform'* along with their lexical scores. Generation task

bar#type#inform ||| *a bar*	0.817394 11.8608 4.06044301054642 0
bar#type#inform ||| *ok a bar*	1.30599 211.551 4.06044301054642 0
bar#type#inform ||| *bar*	2.37433 2.32277 0 0
bar#type#inform ||| *a great bar*	2.99025 16.3807 0 0
bar#type#inform ||| *bar that*	2.65429 1.20412 0 0
bar#type#inform ||| *a bar which*	3.09722 449.996 1.6094379124341 0

There is a clear difference in score distribution which seems to be more well-balanced in case of *'classe'*.

Extending the corpus with additional vocabulary and increasing its size (see chapter 3) evens the probability distributions and diminishes the negative impact of lexical scores, which get more regularity and thus become more informative.

Table 3.15 Tuples for the source word *'classe'*, a rather rare word in the corpus (26 tuples, 98 occurrences). Translation task.

classe ||| *class*	*classe* ||| *ranks*
classe ||| *positions in exactly*	*classe* ||| *attendance*
classe ||| *class-tacitly*	*classe* ||| *ranked*
classe ||| *class-based*	*classe* ||| *middle-income*
classe ||| *new class*	*classe* ||| *attitudes*
classe ||| *flair*	*classe* ||| *establishment*
classe ||| *that ranks*	*classe* ||| *incomes*
classe ||| *classes*	*classe* ||| *elite*
classe ||| *guy*	*classe* ||| *rather*
classe ||| *classifies*	*classe* ||| *makes*
classe ||| *laggard*	*classe* ||| *level*
classe ||| *room for*	*classe* ||| *second*
classe ||| *classroom*	*classe* ||| *NULL*

Table 3.16 Tuples for the source word *'classe'* along with their lexical scores. Translation task.

classe ||| *class*	1.23743 3.43838 1.38629436111989 0
classe ||| *positions in exactly*	1.36798 0.367977 0.693147180559945 3.27714473299218
classe ||| *class-tacitly*	2.6053 0.845098 1.38629436111989 5.31811999384422
classe ||| *class-based*	3.50236 0.403297 0 6.30627528694802
classe ||| *new class*	0.30103 -0 0 0.693147180559945
classe ||| *flair*	6.43114 2.29812 0 3.78418963391826
classe ||| *that ranks*	2.93651 1.63292 0 0.693147180559945
classe ||| *classes*	4.70906 2.82636 0 0
classe ||| *guy*	0.21253 0.234285 0.259108700007725 0.352440639799944
classe ||| *classifies*	3.68827 0.38407 0 5.6021188208797
classe ||| *laggard*	4.77489 0.810318 0 3.04452243772342
...	

3.4.3 CRF generation models

In this section we present our experiments with CRF generation modeling paradigm. Here we assess the performance of CRF models and their potential place in the complete pipeline (Eq. 3.23).

We train the models on the same TownInfo corpus for both English and French and test them on the same testset. The corpus is pre-processed in the same way as in n-gram models training: first concept-to-word alignment is performed using GIZA++ and, second, tuples are extracted according to one of the extraction schemes described in 3.2.2.

In our experiments we use the Wapiti toolkit (Lavergne et al., 2011); it comprises the implementation of several widely used optimization algorithms. For our task we picked two of them: Quasi-Newton algorithm and stochastic gradient descent. Quasi-Newton is a classical learning algorithm which presents a good balance between speed and accuracy. Our choice of gradient descent is dictated solely by the considerations of learning speed, the number of observations and labels being very large (see below).

Also we set up three different scenarios, just like we did with n-gram models: in the first one which we call 'one-to-many' training, we train the model to tag each concept with a full lexical realization which may consist of any number of words. The training instances are created using the modified extraction algorithm ('one-to-many' tuple extraction scheme, 3.2.2). In the second scenario we pre-process the training corpus according to the standard n-gram tuple extraction algorithm, i.e. each source and target phrase is split into sequences of concepts (tuples) and words in strict accordance with the alignment, which produce multi-concept sequences in case of discontiguous concepts. We call this scenario 'many-to-many' training. And finally we expand each concept into a given number of subconcepts, each representing a word in the target phrase; thus a 'lexicalized' version of the source (concept) side has the same number of units as the target side and each subconcept is to be tagged with one word. We called this setup a 'one-to-one' training. Each scenario has its specificities and uses a particular set of additional parameters which we discuss in a respective section below. We evaluate the performance of each setup on the test set using the BLEU metric.

Features

Our choice of CRF modeling paradigm was largely determined by the freedom of feature selection that it offers. For our task, which consists in labeling semantic concepts with their most likely lexical realizations, we consider the features which carry the information about the input itself and the context around it; thus the features that we opt for are the input concepts (we call them *unigram* features) and the ones around (the *context* features), as well as the target side observations. As the maximum number of concepts in a phrase in our corpus is 6 and on average a phrase contains 3 - 4 concepts, all the necessary information lies within this window of concepts.

Thus we use the following features:

- Unigram features- the input concept itself;

- Context features - previous and following concepts (with a maximum window of 2)
- Bigram features - source/target combination as a single observation

As for other model parameters we aim at testing and comparing the following settings:

- Algorithms:
  - Quasi-Newton (QN)
  - Gradient Descent (GD);
- Reordering schemes:
  - ordered: concepts in the training set are ordered (according to the alignment and word order in the training sentence)
  - unordered: no reordering, concepts are presented in the order in which they are given in the training corpus (quasi random)
- Format of tuples:
  - *one2one*: tuples of subconcepts (lexicalized concepts)
  - *many2many*: classical n-gram style tuples
  - *one2many*: tuples created using modified tuple extraction procedure (one concept mapped to several words)

We classify the models according to the features and model parameters they use. The observation strings are build according to a given tuple format and given set of features. For example:

- 1-1 model: previous and the next observations on the source side (concepts 'C') are considered ($C-1$, C, $C+1$)
- 2-2 model: 2 preceding and 2 following observations are considered ($C-2$, $C-1$, C, $C+1$, $C+2$)
- 0 model: only the current observation is considered (C) where C is the current observation (concept)

etc.

It is worth mentioning, that during our experiments we faced the classical problem of trade-off between the complexity of the feature set and the training time. Different setups using different features and different corpus formats result in different number of model parameters and the corresponding training time. For example, the '2-2' model contains approximately 232 million features^5, and takes several days to train. Removing the features reduces training and decoding time, but also results in the performance loss. We discuss the effect of different features and their combinations in respective subsections below.

'One-to-many' model (one concept - several words)

As we have seen, in most cases each source phrase contains several concepts (3.5 on average). Each concept is aligned to several words in the target side. We perform the word-to-word alignment and extract concept-lexemes couples consistent with the alignment to build the training set where each concept is paired with a corresponding phrase. In 'one-to-many' setup each concept aligned with its corresponding lexical realization is treated as an observation. The features that we use in this setup are, first, unigrams with the source side window of 1 in both directions (i.e. the preceding and the next unit). Then we augment the context window gradually, adding one and then 2 words at both sides.

Fig. 3.14 Training instance in one-to-many setup

```
folk#music#inform ||| they play good folk music
chez_sergu#name#inform ||| at chez_sergu
is_a_family_style_hotel#comment#inform ||| and it is a family style hotel
```

By default the concepts in the training set are not ordered; having performed the alignment we are able to order the concepts before feeding the corpus to the learning algorithm. The concepts in the test set are not ordered either which leaves us with two options: label the unordered set of concepts or reorder the concepts before labeling. Both options are tested. We used the reordering rules learned during n-gram training to reorder the test set instances. The results are presented in 3.17.

'Many-to-many' model (n-gram tuples in their original form)

In this scenario the training corpus is built using the native (not modified) n-gram tuple extraction algorithm. We did not perform any reordering. The models were built using Quasi-Newton algorithm and the featureset included window-size-1 unigrams and also bigrams.

[5]We settled on the L2-component of the elastic-net penalty value of 10e-5 for the optimal performance/training_time balance

Table 3.17 BLEU scores for CRF models 'one-to-many' with different configurations, Town-Info English; Quasy-Newton

MODEL	BLEU
QN.1-1.ordered	68.1
QN.1-1.unordered	41.6
QN.2-2.ordered	72.3

In order to test the potential of this setup we also evaluated the performance of the model trained on training corpus containing subconcepts (see section 3.3).

Fig. 3.15 Training instance in many-to-many setup, one2one corpus format

```
B_folk#music#inform I_folk#music#inform ||| they play
I_folk#music#inform I_folk#music#inform I_folk#music#inform ||| good folk music
B_chez_sergu#name#inform I_chez_sergu#name#inform||| at chez_sergu
is_a_family_style_hotel#comment#inform ||| and it is a family style hotel
```

Fig. 3.16 Training instance in many-to-many setup, many2many corpus format

```
chez sergu#name#inform folk#music#inform ||| they play good folk music
at chez_sergu
is_a_family_style_hotel#comment#inform ||| and it is a family style hotel
```

Table 3.18 BLEU scores for CRF models with different configurations (many-to-many), English, GD.1-1

Corpus format	BLEU
one2one	22.1
many2many	0.0

In case of subconcept corpus format, the tuples are smaller and occur more frequently, thus the model was able to derive some patterns and regularities from the training corpus, while pure n-gram-style tuples are in their majority unique and instances seen during training have little chance of occurring in the test set: none of the test instances were correctly labelled by ' many2many' model.

’One-to-one’ model (BIO subconcepts)

In this setup we pre-process the corpus, extending each concept into a given number of subconcepts (equal to the number of words in the target aligned to that concept). We supply an additional indicator of the position of each subconcept relative to the concept scope: B for the first subconcept (B for ’beginning’), I for all others (I for ’inner’).

Fig. 3.17 Training instance in one-to-one setup

```
B_folk#music#inform ||| they
I_folk#music#inform ||| play
I_folk#music#inform ||| good
I_folk#music#inform ||| folk
I_folk#music#inform ||| music
B_chez_sergu#name#inform ||| at
I_chez_sergu#name#inform ||| chez_sergu
...
```

The results are presented in 3.19: *’subconcs given’* denotes a setup where the number of subconcepts corresponds to the one in the reference (i.e. taken from the reference) and in *’subconcs generated’* setup the number of subconcepts is artificially generated according to the fertility model described in section 3.3.

Table 3.19 BLEU scores for CRF models with different configurations (one-to-one), TownInfo English

MODEL	**BLEU**
subconcs given (GD.1-1)	67.9
subconcs given (QN.1-1)	61.1
subconcs generated (GD.0-0)	41.2
subconcs generated (GD.1-1)	50.8

This model shows a relatively good performance even with subconcepts generated using a very basic fertility model. The subconcept format is ideal for sequential tagging algorithms as it allows to take a full advantage of the contextual features and at the same time it manipulates small and thus frequently occurring units used by both the train and the test set (unlike multi-concept units from the previous model).

Adding target features

Out of all CRF models we are mostly interested in one-to-many models that we want to integrate into the general pipeline (see Eq. 3.23) and also in terms of comparison and subsequent combination with n-gram model (which uses the same tuple format). Moreover after the initial experiments these models showed the best performance in terms of BLEU. This is the only model which we decided to test the target features (bigrams) on, as the training time for bigrams does not allow us to test all the models and combinations. So we selected the most informative scenarios (in terms of features) with the optimal balance of performance and training time. This balance is reached with the following feature set: two previous concepts + the concept itself + previously produced target phrase. As our final pipeline combines CRF and n-gram we eliminated the features using the context to the right of the treated concept as it is not compatible with the n-gram model, which uses only the previous context (n-grams). 3.20 sums up the BLEU scores of one-to-many CRF models using different feature sets, including bigrams of source/target observations. We also test these models on the French part of the TownInfo corpus.

Table 3.20 BLEU scores for CRF models with different configurations including the bigram features, one-to-many tuple format, Quasi-Newton algorithm, TownInfo English and French

MODEL	**BLEU en**	**BLEU fr**
-2,-1,0 + B	79.6	54.5
-1,0 + B	78.3	54.6
0-0 +B	65.5	43.1

The inclusion of bigram features into the feature set improves significantly the performance of the models compared to other feature sets. At the same time the increase in training time and computational costs is not negligible. Using target features on a bigger training set might exceed computational capacity even of a very powerful machine.

The results on the French part of TownInfo are worse than on the English part and with different tendencies in BLEU fluctuations: bigrams do not bring any significant improvement in the performance, taking at the same time a considerably larger training time; moreover the difference in performance between the models using different context window sizes is not critical.

Conclusion

In this experimental part of our work we tested multiple CRF model configurations, parameters and feature sets. Our objective was two-fold: first we wanted to closely examine the process of CRF modeling as applied to our data, and secondly, to find an optimal CRF model to be coupled with the n-gram model in the single n-gram/CRF pipeline. Concerning our first objective, we observed the following:

- The performance of one2many CRF models is significantly better than that of many2many or even one2one models. Many2many models yield too many unique observations at training time. As a result the model is not able to make the necessary generalizations and does not learn anything. In many2many scenario the extracted tuples are too long, sometimes a whole sentence represents a tuple, like in Fig. 3.18, which in addition makes context features useless.

- Concerning one2one models, training units are too short – they always cover just one word; thus contextual features even with maximum context window of 5 sometimes do not even cover one entire concept and subsequently are of little use at the inter-concept level. Additionally, these models allow for modifications inside a given concept, unlike in one2many where concepts are learned fixed and a particular lexical representation for a given concept cannot be modified (which is one of the important features of our generation model). These modifications may be beneficial in some, but also may result in disfluent lexical realizations.

Fig. 3.18 many2many training instance, TownInfo (en): the whole sentence is extracted into tuple.

```
metropol#name#inform 980-09-09#phone#inform
bar#type#inform chez_sergu#name#inform  |||
the phone number of the bar chez_sergu is 980-09-09
```

- The use of the union alignment scheme and modification of the tuple extraction paradigm reduced the number of observations and labels in the training set and made it possible to use additional (bigram) features. The performance of models which use bigrams is significantly better than that of unigram models and with an acceptable increase in training time and computation costs.

Concerning our second objective, we found that the best performing model for French is the one which uses 3 features: the concept in question, the previous concept and the target label; in case of English the best performing model is the one which uses a complete feature set, including the context following a given observation. Nevertheless we cannot take these features into account in the complete n-gram/CRF pipeline. Thus we settle on the model, which uses previous context features as well as bigram features, to be coupled with n-gram

3.4.4 Combination of CRF and n-gram generation models

Our general framework (section 3.2.3) allows for combination of different models in the form of finite-state transducers, which enables the overall combined model to use the information from several different sources. Having obtained good results from both CRF and n-gram models we now want to join them into a single graph to make the generation decision even more informed (which in this case is based on several model scores) and select a path which maximizes the combined score. Moreover coupling n-gram with CRF might prove mutually beneficial as both models have different strong points and thus may be complementary.

Thus we want to integrate both CRF and n-gram models into the pipeline along with a reordering model and a language model and then perform an FST decoding on the joint graph in search of the best path. We also want to give each component model a specific weight that we learn via the minimum error rate training, to maximize the BLEU score.

As our pipeline represents a combination of transducers, that is the format we prefer to use for all the components of the pipeline, including the CRF model. To obtain a CRF transducer we transform CRF model format and scores, in accordance with the rest of the pipeline. The obtained transducer is integrated into the pipeline as a *tm2* (Eq. 3.23).

It is worth noting, that the main difference between the CRF sequential labeling in its pure form (as implemented in the Wapiti toolkit) and the CRF model integrated into the n-gram/CRF pipeline remains the decoding paradigm: best path search in an FST graph versus Viterbi decoding in Wapiti. This issue has been discussed in (Lavergne et al., 2011) .

Table 3.21 BLEU scores for the complete pipeline, models: BLM + CRF + TLM; TownInfo English

lexical scores	**model weights**	**BLEU**
no	1.0	79.3
no	MERT	79.5
yes	1.0	67.1
yes	MERT	69.7

There are two things to draw from Table 3.21. First the lexical scores in the original non-extended corpus are detrimental to the overall system performance, and second, the combined BLM/CRF model loses 2.6 BLEU points compared to the BLM model alone. The later observation, though technically correct, requires a thorough examination. We will take a closer look at the outputs of both systems and discuss this issue in the following section.

3.4.5 Additional assessment parameters

Despite the fact that the BLEU score provides seemingly sufficient information about the output quality, there are some elements that pass unnoticed. Obviously the best option would be to ask human judges to evaluate the system's outputs according to some specific criteria. At the same time human evaluation is a costly procedure and cannot be performed during the development stage.

In order to carry out a more thorough analysis of the generated outputs we employ some additional evaluation metrics that provide supplementary information and generally a deeper insight into the output quality. Specifically we are interested in the phenomena which are not captured by BLEU. Among those are:

- Out-of-vocabulary concepts (OOC): input concepts which are not present in the training corpus and consequently have no lexical equivalents (translations). Depending on the decoding options, these concepts may be output by the system in their original form or omitted. In order to detect them easily, we opt for the output.

- Fluency errors (FE): non-coherent phrases, containing grammatical or syntactic errors. Sometimes this errors might not be detected by BLEU, especially in case of syntactic disfluencies.

- Missing words (MW): missing parts of output phrases. This metric helps us detect phrases generated as a result of decoder preference for shorter outputs

- Overall output quality (OOQ): a subjective metric that represents the developers' overall judgment on the system performance (1-5 scale). This metric is a sort of interim human evaluation that we would like to perform during the development period in order to provide additional quality assessments in cases where the fully automatic metric did not always match our own perception of the output quality.

Thus we compare the performance of all component models separately as well as their combination (the n-gram/CRF pipeline) according to these criteria (Table 3.22).

*5/3/2 stands for 5 OOV occurrences, 3 OOV lemmas, 2 non-translated phrases

Remarkably, these estimations do not directly correlate with BLEU. Thus CRF model has a lower BLEU than the n-gram model; at the same time it has much less outputs with missing information and the missing parts of the phrase are not punished directly by BLEU. Thus the combination model is the one offering the best average performance when all the assessment parameters are considered. As a consequence (and despite a higher BLEU score) the n-gram model cannot be considered the best-performing system in our study; instead

Table 3.22 BLEU score and additional evaluation parameters for model components and their combination (n-gram/CRF/TLM)

MODEL	BLEU en	OOC	FE	MW	total errors	OOQ
2-gram BLM	82.01	5/3/2*	7	8	20	4
CRF	78.3	5/3/2*	7	2	14	5
Target LM	75.7	5/3/2*	14	12	31	3
n-gram/CRF/TLM	79.5	5/3/2	7	2	14	5

the combination of n-gram and CRF is retained and used in the upcoming human evaluation (see Chapter 5); this system demonstrated better output quality in terms of the number of non-translated concepts, fluency errors and missing information.

3.4.6 Post-processing

Although coupling n-gram and CRF models significantly diminished the number of non-translated concepts (OOC), there is still a possibility to encounter non-generated semantic concepts in the output text. This is generally acceptable in machine translation, but becomes critical in NL generation. In most cases each concept generates several words, often an entire phrasal unit like an NP or an AdjP. Omitting such a unit would create a gap in the meaning of the phrase and replacing it with a factored concept representation from the source side (which is the default behavior of our pipeline) would make the phrase incomprehensible. In order to diminish the impact of non-generated concepts we implemented an ad-hoc solution: we replace the non-translated concepts with their values. For example:

```
 209-00-38#phone#inform -> '209-00-38'
     in
'the phone number of alexander hotel is 209-00-38#phone#inform'
    becomes
'the phone number of alexander hotel is 209-00-38'
```

Very often non-translated concepts are numbers, dates and prices which can therefore easily be replaced with their values. But in pure concept scenario (without BIO subconcepts) important informative words may be missing. For example:

```
    10|price|inform -> '10'
Cf.: B_10|price|inform I_10|price|inform -> '10 euros'
```

Thus for each specific case (there are not many of them), we add the necessary missing words, like *"euros"* for the concept *"price"*, *"the number is"* for the concept *"phone"*. When applied to our current test set, which has only 5 OOC, the post-processing step does not affect the BLEU score significantly. At the same time it makes the output more comprehensible for average users in cases where the system output does contain OOC.

3.5 Discussion

In this chapter we presented three new generation paradigms, each of which showed a significantly better performance than the phrase-based baseline and demonstrated its potential for the natural language generation task.

First we discussed bilingual n-gram translation framework and its application to NLG: we performed several modifications of the default behavior of the available n-gram modeling tools in order to make them suitable for the generation task; specifically we modified the tuple extraction mechanism to enforce one-to-many tuple extraction, which proved to be the best-performing data format in our setup. The same data format was then used for training CRF models, which made it possible to test both n-gram and CRF models in the same conditions, and later to join them into a unique pipeline in the form of a combination of FST transducers.

We closely examined the performance of both models, as well as their combination, on different data formats; the impact of each parameter has been studied and formally assessed in order to pick the best performing parameter combination.

Bilingual n-gram models showed a remarkable performance on our data; n-gram modeling of joint concept/words pairs proved to be very robust in capturing the essential properties and regularities in the training corpus, which are concept/words n-grams and their probabilities.

CRF modeling paradigm, unlike n-gram models, does not bind input concepts and their lexical counterparts, thus offering more freedom in feature selection and more flexibility at decoding time. Due to these characteristics, CRF models proved to be more efficient in handling out-of-vocabulary input concepts.

The combination of n-gram and CRF models allowed us to bring together the strong points of each modeling paradigm and obtain better results on our data than those of the phrase-based baseline and those of each model in isolation.

Chapter 4

Extending the corpus and the generation model

In this chapter we will take a look at different extensions to our combined generation model that we would like to implement in order to make the user experience in dealing with the system more pleasant and the system-user interaction more efficient. Among those are:

- extending the vocabulary and diversifying the set of syntactic structures available to the system
- adding a modal (pragmatic) system component which would modify generated phrases by incorporating a modal clause corresponding to their discourse type;

To meet our first objective we implement and test several methods that can help automate the process of extending the system vocabulary and augmenting the number of output templates. Specifically, we first replace full words with their synonyms in the target sentences, thus producing new instances for the training corpus, and secondly, we extract paraphrases for contiguous subphrases from the target sentences and build new training instances using these paraphrases.

Concerning our second objective, we use the information about the discourse type of the generated system response; we select a modal expression corresponding to a given discourse type (pragmatic class) and add it to the final output phrase.

This chapter starts with a brief introduction (section 4.1) and a quick overview of the work previously carried out in the field of NLG system extension. Then we move to a detailed description of the two extensions, that we want to implement, namely corpus extension by means of paraphrasing and synonyms integration, and adding a modality component; we discuss the multiple approaches to synonym and paraphrase extraction in sections 4.2.2 and

4.2.3 and the modality/pragmatic aspect of the NLG output in section 4.2.4. Our methodology is presented in section 4.3. There we first address the issue of synonym detection and integration in section 4.3.1., then paraphrasing in section 4.3.2 and finally incorporating the modality component in section 4.3.3. The experiments and results on each step of corpus extension are presented in section 4.4. As before we start with synonyms (section 4.4.1), then we pass to paraphrases (section 4.4.2) and finally to the modal component (section 4.4.3). A small-scale local evaluations of each method are performed at each step in order to estimate their potential for our task. We conclude this chapter with the general discussion in section 4.5.

4.1 Introduction

As we discussed earlier in section 3.3 we created our training corpus from manually built set of templates by replacing the variables (like names, dates and numbers) with values from the database. Thus we got a relatively small corpus consisting of repetitive uniform phrases with very limited vocabulary and low variability in syntactic structures. As it can be seen in Table 4.1. with the total of 34283 words in the English part of the TownInfo corpus, there are only 462 distinct tokens (including such varied units as proper names, numerals, etc). Also it should be noted that the number of patterns (320) includes similar syntactic structures but taking different arguments, e.g. *«There is no $X serving $Y»* and *«There is no $X playing $Z»* are counted as two different syntactic structures, but as it can be seen they are almost identical in their form. Thus the actual number of different patterns is much smaller, just about 30.

As we stressed in the introduction (Chapter 1), diversity of the output may be one of the desirable features of a good NLG system as it improves the user experience and makes the interaction with the system more pleasant. As (Mairesse and Young, 2014) has shown, uniformity and monotony of the system responses has a negative impact on human-system interaction and users in general prefer diverse system outputs, even if they are not the best choice of a surface realization according to the model. His study proved that users are more willing to interact with what he calls an N-best system, i.e. the system which outputs phrases found in the N-best list generated by the model at each turn, which are actually different paraphrases of the same phrase. Also, as we have seen earlier, the size and diversity of the corpus directly affect the generation model itself, specifically probability distributions among generation units (tuples). We discussed this issue in detail in section 3.4.2.

Manual corpus extension is a costly enterprise, at times more costly than building the corpus from scratch, as there are several different surface realizations of a particular concept

Table 4.1 Original Corpus statistics. TownInfo (English and French)

Category	**Num. of units**	
	English	**French**
Size in words	34283	136837
Size in sentences	3151	9000
Number of tokens	462	664
- nouns	205	307
- verbs	81	98
- adjectives	68	76
-adverbs	39	29
Number of patterns	320	284

set that need to be produced in case of paraphrasing. Nonetheless most work on corpus extension has been following this direction: manual paraphrase collection and validation (often using crowd-sourcing). We are not aware of any study on the integration of synonyms into the corpus in order to produce paraphrases.

Outside NLG, both synonym and paraphrase extraction are well established NLP domains with elaborate methodology and a number of available tools and resources which are widely used in various NLP applications. Moreover, automatic extraction methods have been shown to perform at the level close to that of manual annotations (e.g. see (Barzilay and McKeown, 2001)).

In our work we opt for automatic methods of corpus extension. We propose two major extension solutions, which we combine in order to produce a richer and a more diverse training corpus:

1. creating new training instances by means of replacing open class (non-functional) words with their synonyms in existing instances;
2. replacing contiguous subphrases in existing corpus sentences with their paraphrases;

Both solutions are fully automatic and consist in collecting valid synonyms and paraphrases from external resources: a publicly available on-line ontology and general domain parallel corpora. In order to increase the accuracy of the proposed methods we implement automatic filtering procedures which use n-gram language models and also word vector models (section 4.4.1).

Another extension to the generation model itself that we would like to implement is the so-called modal component, i.e. a component responsible for an emotional/sentimental content of generated phrases. It is totally optional and can be removed and added without affecting the rest of the phrase (the principle content). The semantic formalism that we use (with each dialog act represented as a stack of concept-value pairs) allows us to easily integrate a modal clause in a form of an additional concept, without affecting the rest of the dialog act. We use the same techniques that we applied to paraphrase extraction in order to find and retrieve lexical realizations (LR) for modal concepts from an external general-domain corpus. Automatic LR retrieval facilitates the implementation of the modal component as it eliminates the need for manual drafting of modal clauses. We perform several intermediate out-of-context evaluations for each step of corpus extension and for the modal component integration in order to validate the proposed methods. The results are presented in section 4.4.

4.2 Related work: extending NLG training corpora and diversifying the output

4.2.1 Introduction

The problem of diversifying generation system outputs has been gaining considerable interest within the NLG research community, mainly because the users show manifest preference for systems with varied output, which closely resembles that of a human interlocutor. The importance of making systems sound more 'human' has become evident with the increased number of automatic agents and the time we spend communicating with them. Monotone system responses may get annoying after a while and that in turn may divert the user from the interaction with the system in the future.

There has been a number of studies dealing with diversifying the output of an NLG system. All of them focus on paraphrase generation and most of them used crowd-sourcing and other human-expertise sources in order to enrich the training corpus. In (Mairesse et al., 2010) annotators were first asked to provide an utterance matching an abstract description of the dialogue act (concept stack), regardless of the order in which the constraints are presented and then align the attributes and values, e.g. a concept *'food=italian'* to lexical expression *'italian food'*. Thus the annotators performed both text generation and the alignment between the concepts and their lexical realization. The annotations were collected using Amazon's Mechanical Turk. Later in (Mairesse and Young, 2014) the authors extend the previously collected corpus by adding dialogue acts from simulated dialogues. The obtained dataset was manually checked and normalized. The authors report that they ended up with one-two lexical realizations per dialogue act, which means that no paraphrases were used for training and each distinct dialogue act would have the same unique lexical realization each time it is generated. (Mitchell et al., 2014) takes an initial set of language generation templates that have been manually authored, and asks 'the crowd' (a group of untrained annotators) to paraphrase the templates, leaving some designated parts (like predefined concept values) as they are. Also they ask the annotators to evaluate the newly produced templates. This crowd-evaluation is used to help filter the set of new templates that are then presented as candidates to the system developer, who performs further filtering and final integration of the templates into the system. Thus all corpus creation steps, starting from initial templates, through paraphrasing and finally 2-step validation and filtering, are performed manually by a number of untrained annotators and system developers.

Crowd-sourcing, though effective and relatively cheap way to obtain paraphrases, remains unreliable and subject to verification and validation. In case of generation from scratch (Mairesse et al., 2010) it largely depends on the granularity and intuitiveness of the semantic representation. In case of paraphrasing of existent phrases as in (Mitchell et al., 2014), manual validation is necessary to ensure the quality and grammaticalness of the produced phrases and that they are indeed valid paraphrases. Thus both steps - production (paraphrasing) and evaluation (filtering) - require human participation and expertise. It is well known that human expertise, thought still subject to validation, remains more reliable and efficient than automatic methods. Yet there is another aspect that should be taken into consideration and that is the possibility to extend any corpus regardless of the language and domain. Manually produced paraphrases are language and domain specific, and need to be gathered again if the corpus and/or domain changes.

We believe that automatic corpus extension can be performed and is indeed an efficient substitute for manual corpus extension; it requires no additional human resources and expertise, apart from initial evaluation and validation of the method. We go even further claiming that the validation (filtering) can also be done automatically, using the methods which are well-known and widely used in different NLP applications (like machine translation, text summarization, etc). We believe these methods are perfectly suitable for validating newly constructed phrases, be they paraphrases or the output generated by our NLG model. In the two following subsections we briefly discuss the place of synonym detection and paraphrasing in NLG and other NLP domains. Then we continue with the detailed discussion of the work which has been done in different areas of automatic corpus/model extension: synonym detection (section 4.2.2), paraphrasing (section 4.2.3) and modality integration (section 4.2.4).

Integration of synonyms in the training corpus for data-driven NLG models

To our knowledge there have been no attempts to enlarge the size of the system vocabulary by means of synonyms and synonymous expressions. Most of previous research focused on paraphrasing and the more general 'how would you say it otherwise' principle. The idea of replacing words with their equivalents (or closely related lexical units sharing similar semantic content) despite its similarity to paraphrasing, has not been sufficiently investigated and has not been applied to NLG systems. And yet automatic synonyms identification and replacement often achieves higher accuracy and better performance than automatic paraphrase detection. Replacing words with their synonyms, unlike paraphrasing, does not change the syntactic structure of the phrase and thus is relatively 'safe' in terms of its subsequent effects on fluency and grammaticalness of the output. The only concern is the semantic proximity of a given synonym to its reference and how appropriate it is in a given context. In general phrases where words are replaced with their synonyms can be regarded as paraphrases. Cf.:

*It is a **nice** bar **serving** russian food*
*It is a **great** bar **offering** russian food*

These phrases are easier to obtain than valid paraphrases and they have less possibility of being grammatically incorrect or disfluent. We exploit these properties of synonyms in our corpus extension pipeline. We extend the corpus by means of integration of synonyms, replacing the open class words in the original sentences, producing 'paraphrases' of original phrases and adding them to the training corpus.

Integration of paraphrases into the training corpus for data-driven NLG models

As we said earlier there has been a number of studies focusing on collection and integration of paraphrases into the generation model (for the overview of the paraphrase extraction techniques see section 4.2.3).

Besides being successfully applied to NLG, paraphrase integration proved to be beneficial for other NLP domains which handle text generation in some way. (Marton et al., 2009; Mehay and White, 2012; Pal et al., 2014) investigate the impact of paraphrase integration on MT system output quality and MT evaluation. (Barzilay et al., 1999; Das and Martins, 2007; McKeown et al., 1999) apply paraphrases to produce high quality text summaries. (Barrón-Cedeño et al., 2013; Sandhya and Chitrakala, 2011) study the importance of paraphrase identification in plagiarism detection.

Interestingly, in most of these studies paraphrases were obtained and integrated automatically. Yet in NLG domain most work on paraphrase production and integration has been carried out manually. We believe that automatically extracted and validated paraphrases present a reliable alternative to manually constructed (or crowd-sourced) ones, and they are obtained at a much lower cost.

Also previous research on paraphrasing for NLG mainly focused on entire utterances, thus making it harder to extract automatic paraphrases, long sentences being scarce and therefore hard to find in the corpus. In our study we focus on paraphrasing sub-phrases: smaller and more frequently occurring units. We do not take entire phrases and search for their paraphrases; instead we split each sentence into contiguous chunks of different lengths, thus augmenting our chances to find paraphrases for these chunks. In addition that allows us to create different combinations of newly obtained paraphrased subphrases.

4.2.2 Extending the system vocabulary with synonyms

As we said earlier our goal is to diversify a rather monotone corpus vocabulary by adding phrases where the open-class words (nouns, verbs, adjectives and adverbs) are replaced with their synonyms. Synonyms[1] are words that denote the same semantic concept and are interchangeable (to a certain extent) in different contexts and which differ only with respect to their supplemental or peripheral components (lin, 2003). The degree of similarity between the semantic content of synonyms is a subject of debate. Some linguists (Cruse, 1986) argue that synonyms are identical in their meaning, while others (Spark Jones, 1986) distinguish several degrees of 'synonymy': absolute synonyms, partial synonyms, etc. Some linguists claim that there are no synonyms as such; each word in the language has a specific meaning and that is the reason for it to exist[2]. For our study we adopt the definition of Linguistic glossary [LinguaLinks, 2003] cited above, reducing it to only one constraint: two words should be interchangeable within the same context without major modifications in the semantic content of the phrase to be regarded as synonyms. Thus words 'serve' and 'offer' are not considered to be synonyms according to the definition of D. Cruse but in our corpus they are interchangeable in all possible contexts and consequently are regarded as synonyms. Cf.:

Ref.: *They **serve** italian food.*
Syn1.: *They **offer** italian food.*

Here we regard the words and their meaning from the distributional semantics point of view rather than from lexical semantics perspective, as it suits our objective. Notably we use the context information to detect synonyms in the corpus (see section 4.4.1). We aim

[1]There might be some disagreement as to which words should be considered synonyms and to what extent. These judgments are highly subjective and certainly require human expertise. We discuss the issue in section 4.4.1

[2]Certain linguists (Bloomfield, 1933; Bolinger, 1968; Nida, 1958) argued that no true synonyms exist, i.e. that different forms must have different meanings. Thus if a given word exists, it means that there is no other way (or rather other word) to express this particular meaning or semantic nuance.

at automatically finding synonyms for a particular word, thus sparing the human effort of re-writing/extending the templates and integrating them into the training corpus. We test two different methods for synonyms extraction: parallel corpus (PC) method, widely used for paraphrase and synonym detection and a dictionary (ontology) approach, which is an ad-hoc rule-based method, often employed as a baseline and a 'golden standard' for evaluation of data-driven methods. As we said earlier we do not have any knowledge of studies investigating integration of synonyms into the NLG system vocabulary in order to diversify its output. Nevertheless synonym detection and extraction has drawn significant attention of researchers as a part of larger NLP applications: automatic ontology building (Ruiz-Casado et al., 2005), question answering (Riezler et al., 2007), machine translation evaluation (Wong, 2010), information retrieval (Shi et al., 2005) (Wordnet + other ontologies), etc.

Various paradigms for synonym extraction have been proposed and studied. They can be divided into two major classes: rule-based and data-driven. Rule-based methods generally use a knowledge base or a lexical ontology as a source of synonyms and semantically related words. These lexical resources are built manually by experts in the domain and are often used as a «golden standard» for evaluating automatic methods of synonym detection. As their construction is very costly, ontologies are often limited to a particular domain and are somewhat limited in their coverage. The most well-known and widely-used general domain publicly available ontology is WordNet (Fellbaum, 1998). WordNet was used as a major source of synonyms in (Huang et al., 2009; Shi et al., 2005; Varelas et al., 2005) and others. Also, WordNet is widely used by different evaluation metrics as an ontology allowing to search for synonyms: METEOR (Banerjee and Lavie, 2005), MAXSIM (Chan and Ng, 2008), TERp (Snover et al., 2009) and ATEC (Wong and Kit, 2009). Other well-known general domain ontologies are Cyc (Lenat and Guha, 1989) and HowNet (Dong and Dong, 2004). Taking into consideration its relatively broad coverage of the vocabulary in our corpus, we chose to use WordNet to augment the lexicon of our generation models.

Another rule-based (pattern-based) method for synonym extraction was proposed by (Wang and Hirst, 2009; Wang et al., 2012): the author investigates the possibility to extract synonyms and hypernyms from dictionary definitions, suggesting that definitions represent descriptions of an entity which most often use similar terms (synonyms) and broader class names (hypernyms) to refer to that entity.

In data-driven scenario synonyms are extracted directly from the corpus in a supervised ((Hagiwara, 2008), (Hu Fanghuai and Ruan, 2012)) or an unsupervised ((Min et al., 2012)) manner. This methodology is appealing as it does not require human expertise, nor any particular ontology or database; here the set of synonyms can easily be extended by simply adding more corpus texts.

Various distributional similarity approaches have been explored, like that of (Freitag et al., 2005), (Lin, 2004) and (Bollegala et al., 2007), which use the context windows to estimate similarity between words in a monolingual corpus. (Van der Plas and Tiedemann, 2006) proposed a method based on automatic word alignment of parallel corpora in multiple languages. Different translations of the same word in the same context are considered to be synonyms. The authors claim, that their method outperforms the at-the-time state-of-the-art monolingual approach based on distributional semantics.

The combination of different approaches, like in (Wu and Zhou, 2003), where the authors combine monolingual with the parallel corpus approach, yields promising results and outperforms the two methods used separately. (Andrade et al., 2013) uses comparable corpora (CC) instead of parallel corpora which is more abundant and easily available, but at the same time, CC is harder to align which in turn decreases the precision of synonym detection.

For our task we adopt and test the method proposed by (Van der Plas and Tiedemann, 2006) as it showed a relatively high accuracy and is quicker to implement, than a combined approach proposed by Wu. Ontology-based methods remain the most reliable, as they consist in retrieving synonyms which have already been designated as such by human experts. Data-driven methods despite their appeal as «fully autonomous» do not achieve the precision of WordNet, though the recall is much higher. At the same time ontologies are often limited to a specific domain and a particular language. Even general domain ontologies are quite limited in their coverage. In order to balance precision and recall we combine the WordNet based method with a corpus-based one suggested by Van der Plas. We return to the discussion of the trade-off between the precision and recall in section 4.4.1.

4.2.3 Extending the corpus with paraphrases

Replacing words with their synonyms adds diversity to the system vocabulary and variability to the generated output. Nevertheless the basic set of syntactic structures available to the system remains relatively small. Words change with each generated phrase but they stay exactly in the same position within the same syntactic frame and the same context. The obvious solution to the problem of monotony of the corpus is the integration of paraphrases.

Paraphrases are reproductions of the same meaning having different surface forms. Thus paraphrases can be viewed as multi-word synonyms: they are interchangeable in the same context. For example:

Do you like *italian or chinese food?*
Would you prefer *italian or chinese food?*
Would you rather have *italian or chinese food?*

Phrases built by replacing words with their synonyms (as described in the previous section) are indeed paraphrases in themselves. But here we want to go further and paraphrase parts of the phrase which were not affected by synonyms.

Finding paraphrases have long interested the researches. The application domains (text summarization, information retrieval, machine translation, etc.) as well as methods for paraphrase extraction are numerous. (Murata et al., 2005) proposed aligning several dictionary definitions of the same term to extract paraphrases, the definitions being identical in their content, but different in their form. (Wang and Callison-Burch, 2011) uses monolingual comparable corpora built from news articles treating the same topic. (Regneri and Wang, 2012) adds discourse information to the parallel corpus – based paraphrase acquisition algorithm and later also includes Predicate-Argument Structures (semantic roles) in order to detect sub-phrase paraphrases (Regneri et al.). (Barzilay and McKeown, 2001) use multiple translations of the same source text in order to obtain paraphrases.

We chose the technique which is best suited for our task of fragment local paraphrasing and which we applied already to find synonyms. We use word-aligned parallel corpus, but this time we retrieve chunks of words consistent with the alignment and not individual words. The general drawback of this method is that it does not allow to identify large-scale paraphrases containing gaps; only contiguous phrases can be found and extracted. But in our case it is not critical as all phrases to be paraphrased are themselves contiguous chunks of the maximum length of 4 words.

4.2.4 Pragmatic component and modal expressions

After extending the system's vocabulary (section 4.2.2) with synonyms and introducing paraphrases into the training corpus (section 4.2.3), we observed that the responses of the system were more varied and rich in their lexical and syntactic realization. This is a huge step forward in itself towards making the system sound more human. Still there is one important aspect that is missing in the final output of the system: phrases lack emotional component and personal touch; they sound formal and somewhat distant, and yet we have several distinct pragmatic frames in the semantic formalism that we use: *negate, inform, request, select,* etc., which beside their formal surface markers may have a more or less strong relational, personal factor in a human-to-human conversation. This pragmatic/emotional aspect of a phrase is known in linguistic theory as *modality*; in the context of our work we refer to the pragmatic aspect of the NLG system output as "modality component" and to the corresponding lexical and phrasal markers as *modals* or *modal expressions*.

Fig. 4.1 demonstrates the difference between the current system output and the output with an added modal component. This modal component is not crucial for the transmission of the information or for the overall success of the interaction between a user and the system. But it definitely has an impact on the users experience, his perception of the system and his willingness to interact with it in the future. So we envision further extending the model in

Fig. 4.1 Modal component examples

Out 1.*There is no restaurant matching your request.*
Cf.: ***I am sorry**, there is no restaurant matching your request.*
Out 2. *Do you want italian or chinese food?*
Cf.: *Do you want italian or chinese food, **I wonder**?*

order to integrate a modal component into the system output in a form of adjoint modals.

According to the Linguistics Glossary [LinguaLinks, 2003] modality is a facet of illocutionary force, signaled by grammatical or lexical devices, that expresses the illocutionary point or general intent of a speaker, or a speaker's degree of commitment to the expressed proposition's believability, desirability, or reality, as well as his attitude towards the expressed proposition. Linguists distinguish several types of modal expressions: alethic, temporal, deontic, epistemic, doxastic and relational, handling *necessity, time, obligation, knowledge, belief* and *attitude* respectively (Kaufmann et al., 2006), (Palmer, 2001), (Kratzer, 1981). For our purpose we are interested in the modality expressing **attitude**, as it is the type of modality missing in the phrases generated by our system. These modals may include phrases like: 'I am sorry', 'I am happy to inform you', etc. Also we would like to enrich the **fluency** and **interlinking** between propositions by adding connectors, which do not have strong emotional connotations (like, «*also*», «*therefore*», etc.) but are largely used in human-to-human interactions. Thus we distinguish two classes of modal expressions:

1. Sentiment transmission («*I am sorry to inform you*», «*I am happy to inform you*», «*Unfortunately*», etc.). These expressions contain a strong emotional connotation. They are further subclassed into:

    (a) positive («*I am happy to inform you*», «*fortunately*», «*I am glad to tell you*»)

    (b) negative («*I am sorry to inform you*», «*unfortunately*», «*to my disappointment*»)

    (c) neutral («*As you might probably know*», etc)

2. Linking statements: («*Let me tell you*», «*In fact*», «*For your information*», etc). These expressions do not have any emotional content, rather they help link statements and responses one to another and help make the conversation more fluid and natural. They are subclassed,according to the semantic frames that we use, into:
   (a) request, select («*may I ask*», «*I need to know*»)
   (b) inform («*Let me tell you*», «*In fact*», «*For your information*»)
   (c) transitional («*And now*», «*So*,» «*Anyway*»)

We used the paraphrasing method described in section 4.3.2 to obtain more different modal expressions from the corpus.

As we stated earlier, modal phrases do not change the semantic content of generated phrases in any way and thus can be securely added and removed from the output. They are not supposed to interfere with the user's understanding of the phrase, but rather create a friendlier, more relaxed atmosphere during communication. Also in our setup modals are added in a form of an adjunct, thus they do not affect in any way the grammatical structure and coherence of the utterance.

To our knowledge there have been no studies investigating extending the NLG model with an emotional component of this kind. Nevertheless there were a number of studies on the impact of emotional component on the human-machine communication in general (Lee et al., 2002), (Picard and Klein, 2002), and most notably (Petta et al., 2011), which touches upon the possibility of integrating sentiment expression into the NLG module. But in whole these studies treat the problem from an Artificial Intelligence perspective and not a linguistic one; sentiment transmission is reduced to mimics, gestures and voice pitch modifications (prosody). In our study we cast the problem of modality as a *linguistic* or rather *NLP* problem, as sentiments have a lexical and in part syntactic realization ((Pang and Lee, 2008), (Kim and Hovy, 2006)) which gains particular importance within the framework of a dialogue system, especially if the interlocutors cannot see each other. The closest to our objective are studies investigating the possibility of adaptation to a particular user, eg. (Riccardi and Hakkani-Tür, 2005). These models take into consideration user's age, gender, mood and adapt their response accordingly. Normally this happens at the level of dialogue management though, and not language generation.

4.3 Methodology

In this section we present our methodology for corpus and model extensions. As we discussed in the introduction to this chapter, the important points in selecting the corpus extension methods are: quick and easy implementation, availability of training resources (like parallel corpora), scalability and portability to other domains and languages. We describe the methods that we use for corpus extension in sections 4.3.1 and 4.3.2. As for the model extension, namely the integration of a pragmatic (modal) component, we opted for an ad-hoc solution of incorporating an additional modal concept and searching the corpus for its lexical realizations. We discuss our approach in section 4.3.3.

4.3.1 Synonym extraction and integration

For synonym extraction we envision the implementation of two methods: an ontology-based method, which uses a publicly available Wordnet ontology, and a parallel corpus method proposed in (Van der Plas and Tiedemann, 2006).

Wordnet approach

The Wordnet approach is very well suited for our task as the structure of Wordnet allows for a precise and controlable synonym identification. This method proved to be the least error-prone and the most accurate (on our data). The down side though is that Wordnet in its complete form exists only for English. We used an automatic translation tool (Microsoft translate API) to produce synonyms for the French part of the TownInfo corpus (see below).

WordNet (Fellbaum, 1998) is a lexical database of the English language. Open class words (nouns, verbs, adjectives and adverbs) are grouped into sets of cognitive synonyms (synsets); each synset[3] represents a distinct concept (like *'a car', 'to ask', 'kind'*, etc) and comes with a set of lemmas expressing this concept. For example, the synset *'car'* has the following lemmas in Wordnet: *'a car', 'an automobile', 'a machine', 'an autocar'*, etc. Thus synonymous lemmas are grouped into synsets. There are about 117 000 synsets in WordNet. Synsets are interconnected by means of conceptual-semantic and lexical relations into a network of related words and concepts.

We start with selecting the words to be replaced with potential synonyms. First we run Stanford NERtool (Finkel et al., 2005) to exclude proper nouns (names of hotels, bars, streets, etc). After tokenizing the corpus we perform POS tagging using the Brill tagger (Brill, 1992) in order to detect open class words to be searched for in WordNet. For the French corpus we used the TreeTagger (Schmid, 1994) to perform POS tagging. Then we lemmatize the selected words in order to remove plural flexions from nouns and tense markers from verbs as all words in Wordnet are presented in their dictionary form.

[3]Synset – an abstract representation of a concept (ex. *'to serve'*), each synset is represented by a set of lemmas, i.e. words describing this concept. $synset = synonym + set$. Lemmas for the synset *"serve"* include ["serve", "offer", "provide"...etc].

Table 4.2 New training examples built with synonyms obtained from Wordnet (before filtering)

Ref:	*OK , 10 pounds per* ***person***	*and they play* ***ethnic*** *music*
Syn1:	*OK , 10 pounds per* ***man***	*and they play* ***folk*** *music*
Syn2:	*OK , 10 pounds per* ***human***	*and they play* ***indigenous*** *music*
Syn3:	*OK , 10 pounds per* ***individuum***	*and they play* ***national*** *music*
Syn4:	*OK , 10 pounds per* ***head***	*and they play* ***local*** *music*
Ref:	*OK a hotel in any* ***price*** *range*	*Chez Sergu serves chinese* ***food***
Syn1:	*OK a hotel in any* ***monetary value*** *range*	*Chez Sergu serves chinese* ***cuisine***
Syn2:	*OK a hotel in any* ***cost*** *range*	*Chez Sergu serves chinese* ***nutrient***
Syn3.	*OK a hotel in any* ***value*** *range*	*Chez Sergu serves chinese* ***dishes***
Syn4:		*Chez Sergu serves chinese* ***meal***

In our setup we regard each selected word in our corpus as a potential centroid concept in Wordnet; we base our search in Wordnet first on concept names, then on lemma names: if the corresponding concept is found, we retrieve synsets associated with it, if the concept having that name does not exist, we continue to search in the lemma names. If the word is not found in lemma names, it is dropped, and we continue with the next word. Although in our corpus there have been no cases of dropped words, this remains a possibility for other corpora in other domains. For each concept found in Wordnet we pick the first two[4] synsets. Then we extract lemma names and replace them for corresponding words in the initial phrase, producing a new phrase for each lemma name. Thus if there are 3 lemmas in the synset, 3 new phrases are generated, one per lemma name. To produce new phrases we consider all possible combinations of synonyms for all open class words in the phrase. For example, if there are 3 open class words in the initial phrase and 3 synonyms are extracted for each of them, we get 27 new phrases. These cases are rare as on average the original phrases contain 1-2 full words and 3-4 synonyms are extracted for each (see 4.2). For the French part of the corpus we translated all open class words into English, extracted synonyms from Wordnet and translated them back to French. If no translation was found for a particular synonym, it was dropped.

The corpus obtained that way is 15 times larger than the original one (around 45K instead of 3K) but it is quite noisy and needs cleaning and removal of invalid phrases. As you can see in Figure 4.2. A lot of newly extracted words are indeed synonyms of the original ones, but in the given context they might sound unusual.

[4]The first two senses most often are the direct meanings, and not metaphors or slang.

It is worth noting that in many cases the initial meaning of the phrase is not changed even after the integration of a 'bad' synonym; the message is still delivered without major modifications of sense. The only concern remains the fluency of produced sentences. To eliminate the utterances with the most "distorted" meaning we envision implementing automatic filtering (see below).

Parallel corpus approach

This method was proposed and tested in (Van der Plas and Tiedemann, 2006) and it consists in retrieving words having the same translation (and/or translation context) from a parallel corpus (the authors refer to their method as *multilingual alignment-based approach*); the closer the two languages are in the parallel corpus, the better the alignment quality, and thus the higher the precision of the method. The authors claim that their method outperforms the more popular monolingual one (Lin et al., 2003) in both precision and recall as evaluated against WordNet synonyms set which they used as a gold standard. Moreover the authors argue that the coverage of their approach is much higher than that of the golden standard, stating the presence of words that are not found in WordNet. The human evaluation results showed that the synonyms extracted using this method are indeed valid replacements for original words, despite their absence in WordNet. We decided to try this method and see if we can extend further our training corpus with phrases containing synonyms which are not present in WordNet. Additionally this method would spare us the necessity to translate French words in order to extract synonyms from Wordnet for the French part of the TownInfo corpus. We proceed as follows: we extract open class words from our corpus in the same way as we did for the Wordnet method (see section 4.3.1). Then we run GIZA++(Och and Ney, 2003) on a large French-English news corpus to obtain word-level alignment. For each word, we extract all corresponding translations from the aligned corpus. Then for each distinct target equivalent of a given word, we look for all the aligned counterparts in the source, thus getting different translation of the same target word in the source which we consider to be synonyms. We also keep the number of occurrences for each translation pair; pairs with the number of occurrences less than 2 are dropped. For the French part of the corpus we reverse the source and the target sides.

The list of retrieved synonyms include words with completely different meaning. There are two possible explanations for that:

1. parallel corpus sentences are indeed translations of each other, but often not literal translations; fixed expressions (as well as plain expressions) are often translated with their counterparts in the target language which have different surface forms of

Table 4.3 New training examples built with synonyms obtained from the parallel corpus (before filtering)

Ref:	*OK , 10 pounds per* ***person***	*OK a hotel in any* ***price*** *range*
Syn1:	*OK , 10 pounds per* ***man***	*OK a hotel in any* ***money range***
Syn2:	*OK , 10 pounds per* ***human***	*OK a hotel in any* ***currency*** *range*
Syn3:	*OK , 10 pounds per* ***individum***	*OK a hotel in any* ***dollar*** *range*
Syn4:	*OK , 10 pounds per* ***head******	
Ref:	*and they play* ***ethnic*** *music*	*Chez Sergu serves chinese* ***food***
Syn1.	*and they play* ***location*** *music*	*Chez Sergu serves chinese* ***delicacies***
Syn2.	*and they play* ***state*** *music**	*Chez Sergu serves chinese* ***bread***
Syn3.		*Chez Sergu serves chinese* ***snacks***

separate words, having at the same time identical composition meaning (i.e. they are paraphrased).

2. alignment errors

It is difficult to make judgments about the validity of a particular pair - reference word/synonym – in isolation (out of context), so we build new training instances with all the retrieved synonyms and perform filtering on newly constructed phrases instead of just words. We implement two different filters and use them in combination to obtain the final corpus of new training instances that we then add to the original corpus. But first we carry out a human evaluation in order to obtain the golden standard for filter evaluation and also to get an idea on the methods' performance without filtering.

4.3.2 Paraphrases extraction and integration

The method that we chose for paraphrase extraction uses the properties of parallel corpus word alignment in order to extract phrasal units having the same translations. This method was studied in (Barzilay and McKeown, 2001), (Barzilay and Lee, 2003) and (Bannard and Callison-Burch, 2005) Pre-processing steps are similar to those in the synonym extraction scheme. First we extract contiguous blocks of words which do not contain variables (values, like hotel names, addresses, etc) from existing training corpus. They will serve us as bases for potential paraphrases. Each multi-word expression is further split into smaller units to get phrases of different lengths in order to extend the coverage (Figure 4.4). Ex: «*a great restaurant which serves*» can be further split/reduced to «*a great restaurant*», «*a great* *restaurant which*», «*restaurant which serves*», «*great restaurant which*», etc. There are several conditions that we apply:

1. the minimum length of a subphrase is 2
2. at least one word should be an open class word (noun, verb, adjective, adverb) with a single exception of modal verbs (can, may, etc)
3. subphrases are contiguous

Thus subphrases like «*it is*» are discarded. We extracted on average 6.32 subphrases per sentence. In this task we used the same parallel corpus aligned using GIZA++ as we did in case of synonym detection. We traverse the aligned corpus in search of translations for the extracted subphrases. Then it is searched again for the reversed translations of each target phrase found. Thus all source language phrases having the same translation in the target side are considered to be paraphrases[5].

Table 4.4 Subphrases extraction

'I can recommend bar Metropol'	'It is a nice bar on Main road and they serve italian food'
I can recommend bar	*It is a nice bar on*
I can recommend	*It is a nice bar*
can recommend bar	*is a nice*
I can	*a nice bar*
recommend bar	*a nice bar on*
can recommend	*nice bar*
	bar on
	and they serve italian food
	and they serve italian
	....

To build new corpus instances we replace subphrases with their paraphrases creating one new sentence per paraphrase. Thus if two paraphrases are extracted for a particular subphrase, two new sentences are created. If there are several subphrases in a sentence for which paraphrases are found, we create sentences from all possible combinations of subphrases. Then we run the same n-gram filter on the new corpus as for the synonyms. We chose to filter the phrases before handing them down to judges, to eliminate the most distorted and un-grammatical phrases (which are easily detected by the n-gram filter).

[5]a noisy corpus can produce a noisy translation table. The precision of this method depends on the alignment quality and recall depends on the size of the corpus.

Table 4.5 Example output sentences (before and after filtering) initial phrase: a bar which serves

Initially produced paraphrases	**Filtered set of paraphrases**
in the bar that I	*a bar that serves*
a bar where you can buy a	*a bar where I ordered a*
of the bar of you choice which	*a bar where you can buy a*
a bar that serves	*they serve in that bar*
our bar offers great	*the bar owner served them*
a bar where I ordered a	
they serve in that bar	
met in the bar, he	
the bar owner served them	
...	

As we can see in Table 4.5, some of the filtered paraphrases have a different meaning from the original phrase and yet they were accepted by the n-gram filter. That is because they form a grammatically correct sequence inside the sentence.

4.3.3 Modal component integration

There are several ways to integrate a modal component into an NLG system; they differ according to the following criteria:

1. when (at what stage) the modal is integrated: at postprocessing (to already generated output), before training (to the training corpus), at generation time (to the generation graph).

2. modal selection procedure: random (from a predefined set of modals), scored (probability of occurrence of a particular modal in a corpus, relevance), etc.

3. how it is added: specific lexical selection at generation time (modal component affects the structure of the sentence), adjunct clause (no effect on the main sentence), etc.

We base the model selection on the dialog act type (semantic frame). There are currently 14 semantic frames in our corpus: *inform, request, negate, select, hello, confirm, reqmore, repeat, ack, affirm , bye, deny, thankyou, reqalts*. Each frame represents a particular discourse type with a well-defined semantic content. Each of these frames may take a given set of modal expressions to augment its elocutionary force. Thus there are several ways to include these modals into the model at generation time:

1. adding them to the training corpus and thus forcing the model to learn the modals and integrate them into the graph.

2. adding them directly to the graph

3. adding them to the output after decoding (as a post-processing step)

We select the simplest implementation, namely adding the modal to generated phrases at the end of the decoding.

Later we intend to incorporate modals in a form of an additional concept into each input set of concepts produced by a dialog manager (Fig. 4.2). This concept contains the frame name of the dialog act (*inform, request,* etc.), type *'modal'* and the value of *null.* For this setup we keep the value of null, which might be filled up at a later stage when we plan to integrate the modality/user-adaptation component at a higher (decision) level. This newly introduced concept will serve as a slot or a place-holder which may be filled by the dialog manager with the information about the user or his emotional state which will allow the model to determine the right modal to be added. This setup is an undergoing work and is beyond the scope of the present thesis. At this stage we select the modal based on the frame alone and all modals attached to a particular frame have equal scores and equal probability to be chosen.

Fig. 4.2 Modal component integration

Input concepts:	*alexander_hotel#name#inform@1* *209-00-38#phone#inform@2* *10#price#inform@3*
Phrase (default):	*the phone number of alexander hotel is 209-00-38 and the price is 10 pounds*
New phrase:	***null#modal#inform@1*** *alexander_hotel#name#inform@2 209-00-38#phone#inform@3 10#price#inform@4*
Expected phrase:	***Let me inform you that*** *the phone number of alexander hotel is 209-00-38 and the price is 10 pounds*

Augmenting the set of modals

We start with manually crafting one generic modal per discourse type. We then apply the technique for paraphrase acquisition described in section 4.3.2. in order to retrieve all similar expressions for each modal from the corpus. We perform an additional manual checking and filtering (since the set of modals is relatively small, compared to the set of paraphrases, it can be quickly validated manually).

4.4 Experiments and results

In this section we present the experimental setup, evaluation interface and results for each of our extension components.

4.4.1 Synonyms extraction and integration

As we discussed in section 4.3.1 the judgments on whether synonymy holds between any given pair of words is highly subjective. Nevertheless we assume that if a newly constructed phrase conveys the same (or similar) meaning as the reference and replacing a word by another does not prevent the user from understanding the system's response, we consider the word and the replacement to be synonyms. For each word we extracted up to 6 synonyms from WordNet, 3.8 on average, and where applicable up to 10 synonyms from the parallel corpus. For the evaluation setup, we picked at random 100 phrases from our initial corpus[6], both in French and in English. For each phrase we produced up to[7] 20 'equivalent' phrases using 'Wordnet' synonyms extraction methodology and up to 20 'equivalent' phrases using 'parallel corpus' method. We asked 4 evaluators (average users, not linguists), to score the semantic proximity of the generated phrases to the reference on the scale of 0-3. And by semantic proximity we really mean the ability to deliver the correct information, without distorting the sense of the phrase, and not the exact match. Thus words *'bar'* and *'pub'* should be considered absolute synonyms and phrases *'Bochka is a pub that serves beer'* and '*Bochka is a bar that serves beer'* should get the score 3 (despite a slight conceptual difference between them). The task being highly subjective, we also calculated pairwise agreement between the annotators (see Fig.4.3) on 16 test phrases rated by all of them.

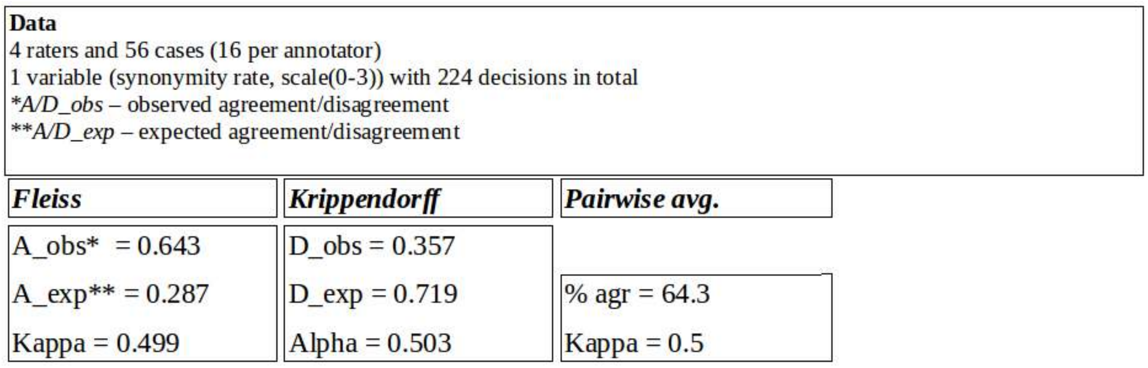
As it can be seen from the agreement statistics (Fig. 4.3), the Kappa is relatively low. This can be explained by the specificity of the task: the judgments on the semantic proximity are very subjective and the scale being 0-3 (and not binary, for example) makes it even harder to reach a high degree of agreement. So in this case we consider Kappa=0.5 to be a sufficient level of agreement for our task. It is also worth noting that the observed agreement is 2.2 times higher than the expected agreement.

Evaluation results are presented in tables 4.6, 4.7 below. We provide the accuracy as well as the average score given by the judges for different word categories. The two tables provide two different accuracy scores: the first table contains *inclusive* accuracy scores, i.e., phrases given the score of 2 or 3 are considered to be similar to the reference; the

[6]As a significant number of phrases in our corpus are similar in structure, but with different variable values, which we do not consider in our synonym extraction scenario, this number should be representative enough

[7]The actual number of equivalents for a given sentence depends on the number of open-class words in a given sentence as well as the number of synonyms found for each OC word.

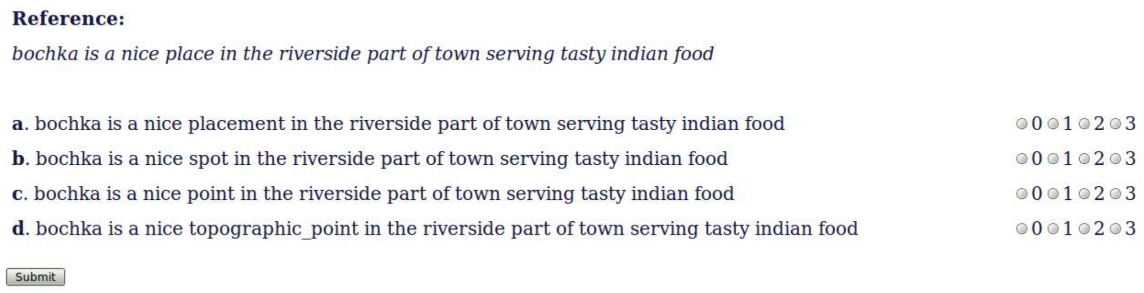
Fig. 4.3 Pairwise agreement statistics between the annotators ((Cohen, 1968; Fleiss, 1971; Krippendorff, 1980))



Data
4 raters and 56 cases (16 per annotator)
1 variable (synonymity rate, scale(0-3)) with 224 decisions in total
**A/D_obs* – observed agreement/disagreement
***A/D_exp* – expected agreement/disagreement

Fleiss	***Krippendorff***	***Pairwise avg.***
A_obs* = 0.643	D_obs = 0.357	
A_exp** = 0.287	D_exp = 0.719	% agr = 64.3
Kappa = 0.499	Alpha = 0.503	Kappa = 0.5

Fig. 4.4 Evaluation interface: synonyms



Reference:
bochka is a nice place in the riverside part of town serving tasty indian food

a. bochka is a nice placement in the riverside part of town serving tasty indian food ○0 ○1 ○2 ○3
b. bochka is a nice spot in the riverside part of town serving tasty indian food ○0 ○1 ○2 ○3
c. bochka is a nice point in the riverside part of town serving tasty indian food ○0 ○1 ○2 ○3
d. bochka is a nice topographic_point in the riverside part of town serving tasty indian food ○0 ○1 ○2 ○3

second table contains *exclusive* accuracy, i.e., only phrases given the score of 3 are viewed as being synonymous to the reference. PC and WN stand for 'Parallel Corpus' and 'Wordnet' respectively.

Table 4.6 Inclusive accuracy by word category.

Word category	A (PC)	A (WN
all words	0.51	0.64
nouns	0.46	0.49
verbs	0.77	0.82
adjectives	0.53	0.55

As it can be seen from the result tables, there is a considerable difference in accuracy between different word categories: accuracy for nouns is less than 0.5 for both Wordnet (WN) and parallel corpus (PC) methods in inclusive and less than 0.3 in exclusive scenarios, while the scores for verbs reach 0.82 and 0.71. One possible explanation for this might be that nouns tend to be more polisemic, than verbs or adjectives, hence they show more divergence

Table 4.7 Exlusive accuracy by word category.

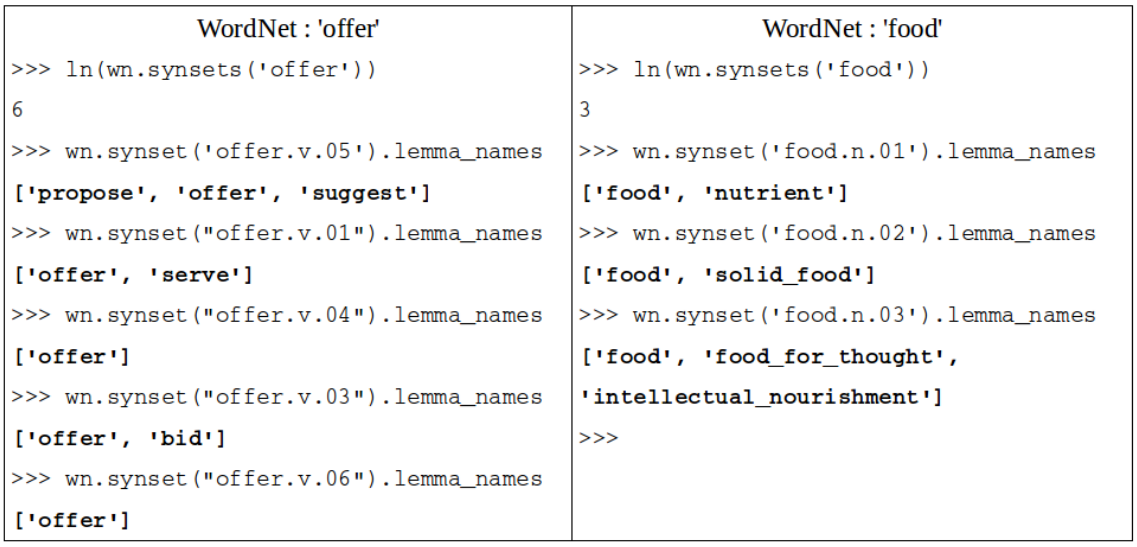
Word category	A (PC)	A (WN)
all words	0.39	0.42
nouns	0.23	0.26
verbs	0.69	0.71
adjectives	0.45	0.47

in meanings for the same lexeme. Unlike verbs or adjectives, which are more restricted in their meaning, nouns tend to develop metaphoric or abstract (figurative) senses.

Table 4.8 Average score per word category. PC and WN stand for ’Parallel Corpus’ and ’Wordnet’ respectively

Word category	Av. Score (PC)	Av. Score (WN)
all words	1.53	1.89
nouns	1.19	1.23
verbs	1.98	2.11
adjectives	1.51	1.54

Fig. 4.5 WordNet examples



WordNet : 'offer'

```
>>> ln(wn.synsets('offer'))
6
>>> wn.synset('offer.v.05').lemma_names
['propose', 'offer', 'suggest']
>>> wn.synset("offer.v.01").lemma_names
['offer', 'serve']
>>> wn.synset("offer.v.04").lemma_names
['offer']
>>> wn.synset("offer.v.03").lemma_names
['offer', 'bid']
>>> wn.synset("offer.v.06").lemma_names
['offer']
```

WordNet : 'food'

```
>>> ln(wn.synsets('food'))
3
>>> wn.synset('food.n.01').lemma_names
['food', 'nutrient']
>>> wn.synset('food.n.02').lemma_names
['food', 'solid_food']
>>> wn.synset('food.n.03').lemma_names
['food', 'food_for_thought',
'intellectual_nourishment']
>>>
```

Fig. 4.5 shows the listings of WN synsets for the concepts *’food’* and *’offer’*. *’Food’* has a figurative meaning *’food for thought’* which is quite different from its original one, while all senses of *’offer’* are related and its lemmas can be used interchangeably.

Another observation concerning the results is that WN provides in general more precise and relevant synonyms than the PC approach which is understandable since WordNet was built by human experts. At the same time the fact that WN and PC scores are rather close allows us to highlight the potential of PC approach as a cheap, reliable, language- and domain-independent alternative to WordNet.

Filtering

Human expert filtering

As we have seen in the previous sections, both Wordnet and Parallel corpus approaches produce a lot of invalid or incorrect synonyms. In order to be able to safely add the newly constructed phrases to the training corpus we need to find a way to filter out invalid ones. One obvious way to do it is to use human judgments about the proximity of the newly built phrases to the reference. Thus if *'american_indian'* instead of *'indian'* in *'... serving indian food'* was given the score of 0 by all the annotators, all phrases containing *'american_indian'* were dropped from the corpus. We established that only phrases scored 2 and 3 by at least 2 annotators were to make their way into the new training corpus. Though human evaluation remains the most reliable and efficient way to filter out incorrect phrases, it is too costly to be carried out on the entire corpus; moreover it needs to be performed each time we decide to add new phrases to the training corpus. A solution would be to resort to automatic filtering and keep human judgments as a golden standard to evaluate the performance of automatic filters.

Automatic filtering

Using evaluation results to filter out incorrect phrases is the most straightforward solution as it provides a reliable 'human' filter for the constructed phrases. But the evaluation is performed on a small sample of the corpus and filtering needs to be performed on the whole newly built corpus. In addition, using human expertise to filter the synonyms (even if it was supposed to be used for evaluation in any case) strips our approach of its «fully automatic» status. Secondly it makes the approach not portable to a new corpus or a new NLG system, if we were to resort to human expertise each time we build a new corpus (or extend the existing one). Thirdly there exist some well-established techniques, like n-gram language modeling, to filter out un-grammatical phrases; moreover, the rapid development of (and a new wave of interest in) distributional semantics (DS) provides new methods for similarity evaluation and along with it for synonym filtering. We take advantage of both 'old' and 'new' methods and use them to implement two automatic synonym filters: one based on the distributional semantics (DS) similarity scoring, specifically on word vectors, and another one based on n-gram language modeling. We evaluate the performance of both filters separately against the golden standard human judgments in terms of precision and recall. Finally we run the double-filtering pipeline on the newly obtained corpus of synonymous phrases to produce the final training corpus. We proceed as follows: first we run a DS filter; as multi-word expressions cannot be handled directly (as they are) by vector approach, we fall back to an n-gram LM filtering to score multi-word synonyms.

Distributional semantics filter

Distributional semantics (DS) is a grounded and mature NLP field. It has been around for a while and has long been studied for its capacity to detect similar words and phrases. Recently it has been revived with the introduction of word vectors (Mikolov et al., 2013): real vectors that represent a word an in n-dimensional space.

And though previous studies showed that DS methods in many cases are unable to distinguish between different types of semantic proximity, such as antonyms, hyponyms, hypernyms and synonyms (Van der Plas and Tiedemann, 2006), more recent work revealed its potential to closely model different semantic relations, including synonymy with a significantly higher accuracy (Mikolov et al., 2013), (Dao et al., 2013), (Maas et al., 2011).

We used an open-source tool Word2Vec[8] to generate word vectors for extracted synonyms and references. The model is trained on Gigaword 5th edition(Parker et al., 2011), Brown corpus (Francis, 1965) and EnWiki(Reese et al., 2010) using skip-gram algorithm with the vector size of 300 and the window of 10 words. We compare the vectors by calculating pairwise similarities between a reference vector and synonyms vectors for each set of synonyms. For each set of vectors we determine a threshold by taking the average similarity score for each set. Thus we keep the words whose vectors are sufficiently close to the reference vector and remove the rest.

N-gram LM filter

Another filtering method that we adopt uses the 5-gram language model built on the Web corpus (Brants and Franz, 2006). N-gram language models are widely used for estimating fluency and validating the output of various NLP systems. It is particularly important for tasks involving generation of phrases in natural language, such as machine translation and natural language generation. In our case we validate the phrases generated by replacing the words with their synonyms.

[8]https://code.google.com/p/word2vec/

In our setup, apart from the synonym word, the phrase remains identical to the reference. So the LM estimations for the rest of the phrase would not change. Our objective is to evaluate, how well the synonym fits into a given context. To avoid problems of sparsity and to make LM estimations more reliable we decided to take just the part of the phrase containing the synonym, i.e. the word itself with the window of two words before and after it. Thus we really evaluate 5-grams containing the synonym.

Ex: *Botchka is a nice bar and they play* ***ethnic*** *music .*
Botchka is a nice bar and they play ***folk*** *music .*

According to our filtering scheme, only the part *'they play folk music'* is scored and not the rest of the sentence, which is likely to provide a more robust estimation. Here again we determine a threshold by taking the average score of a set of synonymous phrases and remove phrases with the score below the threshold.

Filter evaluation

We evaluate the filters by comparing their output to the human judgments obtained earlier. We used the same set of phrases that we presented to the judges. The results are presented in Table 4.9.

Table 4.9 Filters evaluation

Filter	Precision	Recall	F-score
DS	0.61	0.49	0.54
LM	0.78	0.71	0.74

As we see DS filter has a low recall (0.49) but a fairly high precision (0.61). This can be explained by the fact that many of the valid synonyms according to Wordnet have a low similarity according to the word vector model, thus they are dismissed. The n-gram filter on the contrary has a significant recall. This is probably due to the size of the Web corpus which is bigger than Gigaword and also it takes into consideration the context, while DS filter treats separate words. Thus *'serve'* and *'offer'* have a low similarity score according to the word2vec model (0.2122) but in the context of 'a bar' like in *"and they serve italian food"* (cf. *"and they offer italian food"*), their meaning is identical; *'offer'* was dismissed by DS filter, while N-gram filter was able to capture the similarity.

Discussion

We achieved our objective of extending the training corpus, increasing the vocabulary of our generation system and thus diversifying its output. Specifically we augmented the size of the corpus by 6: from 3K to 22K. We applied the techniques from synonym extraction which are widely-used in different areas of NLP, notably in information retrieval for query expansion and in plagiarism detection, but which have been generally neglected in NLG. The techniques are automatic and does not require human expertise. Wordnet method cannot be considered fully automatic as it requires a hand-crafted ontology, which is language specific and sometimes also domain-specific. General domain ontologies of that kind exist only for a small number of languages. Moreover Wordnet has somewhat limited coverage. Parallel corpus method on the contrary is portable and can be implemented regardless of the domain: domain specification is handled at the corpus level by picking a particular corpus of a particular domain. Wordnet method showed a relatively high precision, while the precision of the PC approach leaves much to be desired. On the contrary the recall in PC is high. Nevertheless, low precision is compensated by automatic filtering that we apply to the extracted sets of synonyms.

Table 4.10 Corpus statistics TownInfo (English) extended with synonyms, before and after filtering

Corpus parameters	**before**	**after**
Size in words	574619	205839
Size in sentences	40650	18906
Number of tokens	827	547
- nouns	457	240
- verbs	126	96
- adjectives	116	88
- adverbs	55	45

Table 4.11 Corpus statistics TownInfo (French) extended with synonyms, before after filtering

Corpus parameters	**before**	**after**
Size in words	712119	343512
Size in sentences	51909	23786
Number of tokens	1181	689
- nouns	534	288
- verbs	159	106
- adjectives	99	89
- adverbs	62	47

The filtering is rather crude, it does not attain sufficient level of accuracy compared to human filtering: it does occasionally eliminate valid synonyms and allows for integration of bad ones. A more sophisticated approach to filtering is needed. Still automatic filtering has a great potential as the evaluation results prove.

Rule-based synonym extraction methodology, despite rather good results on our corpus, is not portable to another language or to a more specific domain. The PC method on the contrary is easy to implement and apply (provided the availability of the in-domain parallel corpus) to any data.

A more elaborate implementation of the PC method would be to account for the context in which a given word appears. As of now we treat the words as separate units and do not take context into consideration when retrieving synonyms for a particular word. On the other hand both filtering methods use contextual information, which should compensate for the lack of context and help remove the most inappropriate words. Still we believe, that a proper implementation of synonym extraction would be to retrieve the words along with their contexts and base the decision on their proximity on multi-word sequences. We leave it for future work.

4.4.2 Paraphrase extraction

We apply our paraphrase extension methodology to the corpus that we obtained as a result of adding synonyms (Table 4.10), as this corpus should provide a richer base for paraphrasing. We apply the paraphrasing algorithm described in section 4.3.2 to the English side of the corpus.

Table 4.12 Statistics on extracted paraphrases and the newly created sentences (English)

Category	Num.of units
Subphrases per phrase (avg)	6.1
All subphrases	1447
Paraphrases per phrase(avg)	5.8
Phrases with no paraphrases found	288
All paraphrases (before filtering)	3996
All paraphrases (after filtering)	2139

The number of extracted subphrases varies greatly from sentence to sentence, depending on its length and the number of open-class words that it contains. For example a sentence like *'It is on the Main Road'* will get much fewer paraphrased variants than a sentence *'The phone number of Chez Sergu is 234-45-45 and it is a nice restaurant in the central area of town'*.

Table 4.12 gives an overview of the number of paraphrases that we obtain for English. As we can see, even after extending the original corpus with synonyms the number of distinct subphrases is relatively small. This is due to still somewhat monotonous nature of our corpora.

There were phrases for which no paraphrases have been found. The maximum number of paraphrases found was 14. Obviously not all extracted paraphrases were valid and the valid ones do not always yield a fluent sentence when placed in the context of the original sentence. As in case of synonyms we applied the n-gram filter, but this time the entire sentences were considered, not just the paraphrased parts. We also used manual evaluation for additional filtering later on.

We did not carry out the human evaluation this time, like we did for synonyms, due to the organization costs; instead we performed a manual filtering where we selected valid paraphrases and removed the rest. Due to the lack of time, we leave the experiments with paraphrasing in the French corpus for future work.

Discussion

We achieved our objective of extending the corpus with automatically obtained paraphrases. The evaluation showed that automatic paraphrase acquisition can be a reliable alternative to crowd-sourcing and deliberate manual paraphrase creation. It is less expensive in terms of time and effort. At the same time the quality of obtained paraphrases remains subject to validation. It depends on many factors, among which are, as in our case, the size of the parallel corpus, the quality of the alignment, subphrase extraction algorithm and the robustness of the filtering mechanism.

Table 4.13 Corpus statistics TownInfo (English) before and after extending with paraphrases

Corpus parameters	**before**	**after**
Size in words	205839	466129
Size in sentences	18906	42198
Number of tokens	547	588
- nouns	240	261
- verbs	96	103
- adjectives	88	96
- adverbs	45	52

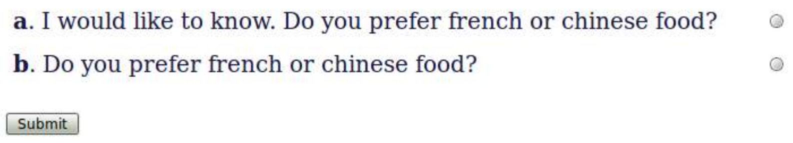
The split into subphrases in our setup is not linguistically motivated, i.e. the boundaries of the subphrases do not match those of standard linguistic category groups (like VP, NP, etc). The same is true for their paraphrases, extracted from the corpus. That is why quite a number of produced paraphrases of original sentences are discarded during filtering, as being un-grammatical or disfluent. Nevertheless we managed to obtain quite a number of new syntactic constructions which were not present in the original corpus.

4.4.3 Modal component integration

To assess the impact of adding the modal component to the generation systems we perform a very basic evaluation with just several users, presenting them with two variants of the same phrase and asking them which one they prefer. The users are supposed to take into consideration not just the fluency (or the surface form of the new phrase) but also relevance and appropriateness of the modal in that particular context.

We presented the evaluators with a set of phrases (30 per evaluator) generated by our best-performing generation model (see chapter 3) trained on the original corpus and their counterparts containing modal components.

Fig. 4.6 Evaluation interface: modality clause added at generation time


a. I would like to know. Do you prefer french or chinese food?

b. Do you prefer french or chinese food?

Table 4.14 Evaluation of the modality component

Evaluator	+ modal/ratio	- modal/ratio
User 1	22/0.74	8/0.26
User 2	25/0.84	5/0.16
User 3	22/0.84	8/0.16

As evaluation results in the table 4.14 show, the users give larger preference to phrases containing a modal, though there are cases when the modal was considered inappropriate. In some cases the users signaled that some modals made phrases too heavy and even too formal. It was also signaled that repeating modals overload the conversation, at the same time a complete absence of modals dries the conversation out. A compromise solution remains to be found, like adding modals occasionally, or a more careful modal selection which helps to avoid repetitions and similar modals in the same conversation act.

Discussion

In addition to extending the vocabulary and diversifying syntactic structures we added a modal component responsible for emotional/pragmatic content of generated phrases. We obtain a set of modal expressions for each specific modal group (frame) automatically and add them to the generated phrase after decoding as a post-processing step.

According to the preliminary evaluation results modified phrases sound more «human» and are more user-friendly according to the evaluators. All judges in the present evaluation setup have shown an overwhelming preference for utterances containing the modal component. Though in some rare cases the added modals were signaled not to fit well to the main phrase (due to the random modal selection procedure).

We obtain the modals automatically, thus adding another (emotional) dimension to our model without handcrafting the corresponding patterns nor involving any kind of additional human resources. As the modals are often standalone expressions which can be added and removed at will without any impact on the grammaticalness of the phrase, adding a modal component is a relatively safe extension in terms of subsequent effect on the fluency. They are always 'external' to the original phrase and always either precede it or are appended to the end. The only concern remains the appropriateness of a given modal in a given context. This cannot be determined automatically (by means of a filter) and needs to be signaled by a user.

There is a number of issues that are not addressed in our approach: in this setup we do not take into account user's feedback, nor the history of the conversation; the choice of a modal depends solely on the pragmatic frame, no other semantic components are taken into consideration.

4.5 Conclusion

As we discussed throughout the chapter, diversity, variability and 'humanlikelyness' are of great importance when it comes to interaction between the user and the dialog system. Repetitive formal responses might get annoying and divert the user from the intention to interact with the system in the future. Thus diversifying the system's output and making it «user-friendly» is crucial in the conception of an NLG model. In this chapter we presented several solutions that we implemented in order to make the user-system interaction more pleasant and efficient.

1. Integration of synonyms into the training corpus is a simple, yet an effective solution which has not been fully explored and applied to NLG up to now.

2. Paraphrasing is a well-known technique used in NLG largely in the form of crowd-sourcing or manual paraphrasing, yet it is also very effective and less expensive when

implemented as an automatic process which does not require human intervention (apart from evaluation).

3. Paraphrasing parts of a phrase, and not the entire phrase, creates smaller reusable subphrases, thus allowing for different combinations of subphrases, which in turn adds even more variability and further augments the size of the training corpus.

4. Modality/emotional component has not yet been implemented at the level of NLG in a dialog system, despite it's largely linguistic nature which might and indeed does have a surface realization in a form of a modal clause. As human evaluation showed, adding a modal component to the system makes it more appealing to end-users, as opposed to the baseline.

5. Apart from the modal component, which is language- and task-specific, all the steps in corpus extension and enrichment are fully automatic and do not require any human intervention (except for the evaluation); they are universal and can be used to enrich the training corpus for any domain and any language.

We have created an extended and diverse training corpus which can be used not only for training other NLG models, but also in language understanding and other NLP domains.

Chapter 5

Human Evaluation

In this chapter we describe our setup for human evaluation that we carry out in order to assess our two best-performing generation systems for English. We start with the ground for performing an extended human evaluation and a brief description of the systems we plan to assess (section 5.1). In section 5.2 we introduce the evaluation framework that we designed, as well as the selected estimation criteria. The judges' profiles, evaluation interface and the overall deployment of the evaluation process are outlined in section 5.3. Section 5.4 presents the results and examines the correlation between the automatic and the human evaluation scores on our data. Finally we end this chapter with the error analysis in section 5.5 and the discussion in section 5.6.

5.1 Introduction

While automated evaluation metrics provide useful information for comparing different systems during development stage, the final estimation of the systems' quality is left to human judges. Their feed-back is indispensable for evaluating such system properties as naturalness and diversity of the output which are hard (if at all possible) to assess by means of automatic metrics.

Specifically in our final evaluation setup we do not just want to assess the proximity to a particular reference in terms of n-gram matches, which can be done using the BLEU metric like we did in intermediate evaluations during the development stage. Rather we would like to go further and get human assessment in terms of:

1. overall fluency/grammaticalness

2. user-friendliness/naturalness

3. diversity
4. informativeness

all of which could not be measured with automatic metrics. For example, the BLEU score captures local similarities between the reference and the generated phrase, but it can completely overlook the overall non-grammaticalness of the phrase. At the same time it can downgrade well-formed and informative system outputs if they differ significantly from a given reference (or references).

We would like to compare our two final NLG systems (combined n-gram/crf model for English and its extended version) with template-based in-house generator and also between themselves: specifically in the later case we would like to see the effect of the corpus extension and the presence of the modal component on the user experience during the interaction with the system.

There are two possible evaluation scenarios that may be envisioned in our case. In the first one - which we refer to as intrinsic evaluation scenario - the users are presented with generated phrases out of context and are asked to rate these phrases in terms of naturalness, fluency and informativeness (see section 5.2.1). In the second setup – which we call extrinsic evaluation scenario - the NLG models are incorporated into the dialog system and the users interact with it in a continuous dialog (section 5.2.2). This evaluation setup would allow to observe and assess the performance of the generation module in correlation with other components of the dialog system. Each scenario uses a particular evaluation protocol and is conceived to provide an insight into specific system characteristics and evaluate specific properties.

Unfortunately the deployment of the extrinsic evaluation is a costly enterprise and requires a full functioning dialogue system and a user interface. Due to time constraints we decided to perform a thorough and extensive intrinsic evaluation and leave the extrinsic setup for future work. Nevertheless we discuss the procedure protocols of both in a corresponding subsection of section 5.2 below.

5.2 Evaluation protocol

As we said in the previous section, we want to assess the following output properties: naturalness, fluency, informativeness and diversity of the output.

Each criterion assesses a particular aspect of the generated phrase. Naturalness is defined as the ability of the system to output phrases indistinguishable from those generated by humans. Here we imply the use of modals (modal component), forms of politeness, informal conversational style of the output, flexibility in the selection of syntactic constructions and lexical units.

Fluency is the most important criterion that we want to assess; it reflects the ’well-formedness’ and the general grammaticalness of the output. It might seem somewhat close to naturalness; yet a phrase can be well-formed from the point of view of grammar and sound artificial or too formal.

Informativeness denotes the presence of all the information (concepts) received from the dialog manager in the generated output. Normally this criterion is captured by automatic metrics like BLEU in case of comparing the output to multiple references (if both the output and the reference have concepts expressed in the same way). Otherwise it takes a human expert to make a judgment on the completeness of the phrase and the presence of all the necessary information.

Fluency, naturalness and informativeness can be measured at the phrase level and thus make the object of the intrinsic evaluation (section 5.2.1). Diversity reflects the ability of the system to output different (diverse) phrases and is determined by the three main factors: the size of the vocabulary, the number of syntactic structures and the mechanism of their selection. The diversity of the output is difficult to capture and quantify at a phrase level. This criterion is evaluated at the dialog level or after a prolonged (multi-phrase) interaction with system. We discuss the extrinsic evaluation scenario in section 5.2.2 below.

It is important to note that real-world dialog system users are not always native speakers of the language used by the system; for example they might be forced to communicate with it in English, if it does not operate in their native language. We would like to see the performance of the system from the point of view of both groups of users - native and non-native speakers - and assess their ability to understand and use the information provided by the system.

5.2.1 Intrinsic evaluation

In this scenario we ask the users to assess the usability of the generated phrases out of context, i.e. as isolated conversation units.

Evaluation setup

The judges are presented with outputs of several systems, including the in-house rule-based generator and asked to score each output on the 1-5 scale according to the three criteria that we defined above: fluency, informativeness and naturalness. The semantic content of each phrase in the form of a concept stack is also given to the evaluators, which lets them adequately assess the informativeness of the outputs. After each turn containing several outputs for the same semantic representation the judges are asked to rank the systems according to their preferences (based on any criterion or just their personal taste). Thus each turn allows us to collect two types of information concerning the users' preferences: their judgments on the quality of each particular phrase produced by a given model and their preferred one(s).

Systems description

For the evaluation we selected three NLG models: our best-performing combined n-gram/CRF model, an extended version of the combined model and the template-based generator.

The combined n-gram/CRF model represents a weighted combination of bilingual n-gram, CRF and target LM models built on our original corpus (the full description is given in Section 3.4.4). The extended version of the combined model is the same basic model built on the extended corpus - our original corpus enriched with synonyms and paraphrases. Our first objective is to compare the basic (non-extended) model performance with that of the rule-based model: we want to make sure that our data-driven modeling approach is comparable in terms of fluency and the overall quality of the output with a hard-coded template-based model. Our second objective is to validate our corpus extension methodology by comparing the outputs of the extended model with the output of the basic n-gram/crf model.

We use the same test set that was used for previous automatic evaluations (230 dialog acts in total). This will allow us to compare the results and assess the correlation of BLEU with human judgments on our data.

User profiling

We would like to obtain the assessments from two groups of evaluators: native and non-native speakers. The reason for this is that in the real world, the dialog systems, especially the ones providing tourist/reservation information and services, are often used by non-native speakers and should therefore be easily accessible and comprehensible for all potential users. Thus the judges' sample should be diverse and representative of the target audience in terms of such profile parameters as age, gender and the level of English.

We also are aware of the complexity of the task for untrained evaluators, specifically of a narrow distinction between the criteria in some cases and the subsequent difficulties in a proper scoring. Thus we reduce the number of dialog acts in each run to 10, which makes 30 utterances in total per user per run. Users are allowed (invited) to make several runs, if they feel comfortable with it.

5.2.2 Extrinsic evaluation

In this scenario the judges enter in a direct interaction with the dialog system and simulate a dialog within the context of our domain (tourist information/booking). Specifically the users are asked to make a reservation in a place of their choice. A number of instructions may also be provided to the users as to what information they might like to ask for and how they should build their dialog with the system.

In this case the assessment concerns the overall success of the interaction and whether the user was happy with it or not. Also it is intended mostly for the assessment of our extension methodology and whether the 'extended' systems sound more appealing to users than the original ones.

As this kind of evaluation is more expensive in terms of deployment than the previous intrinsic one, we cannot perform it within the framework of our current study. We leave it for future work.

5.3 Evaluation deployment

In this section we present the evaluation setup: the interface, judges profile, evaluation environment and timing.

5.3.1 Participants

We start off by selecting the participants for the evaluation campaign. We base our selection mainly on their native language and their proficiency in English. Our objective is to gather a diverse and representative sample which would include several age groups and both genders. The participants are divided into two major groups: native and non-native speakers. Non-native speakers are further divided based on their English level into advanced and intermediate English speakers. We want to make sure that each group has a sufficient number of users (for the purpose of subsequent analysis of the results). We do not perform a fine-grained groupings according to any other criteria, like age, gender or social status, as in our setup these characteristics are less pertinent and informative, though such analysis is possible and would probably provide a deeper insight into the specificities of the system-user interaction.

Thus we end up with 56 participants in total:

- 6 native speakers,
- 17 intermediate and

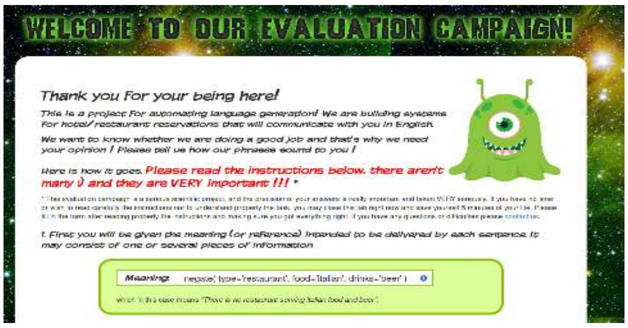
- 33 advanced speakers.

The distribution is not well-balanced, but we consider it to be sufficiently representative of each users group. The geographical profile of non-native speakers is rather large: both advanced and intermediate users combined represent 12 countries.

Each dialog act (turn) in the test set has been scored by at least 2 judges (3 in many cases). We use a sample of 30 phrases scored by 3 judges to calculate the inter-annotator agreement in each group (see section 5.1). We discuss the agreement between the users in greater detail in section 5.4 below.

5.3.2 Interface

As we said earlier, our user sample consists of people residing in different countries. The easiest way to reach out to this multinational group of users is to build a web-based interface. Our overall goal when building the interface was to make it as simple and intuitive as possible, providing at the same time maximum information and explanations concerning the task at hand. The web-based evaluation interface is presented in Figures 5.1, 5.2 and 5.3.



WELCOME TO OUR EVALUATION CAMPAIGN!

Thank you for your being here!

This is a project for automating language generation! We are building systems for hotel/restaurant reservations that will communicate with you in English.

We want to know whether we are doing a good job and that's why we need your opinion ! Please tell us how our phrases sound to you !

Here is how it goes. Please read the instructions below, there aren't many ;) and they are VERY important !!! *

* This evaluation campaign is a serious scientific project, and the precision of your answers is really important and taken VERY seriously. If you have no time or wish to read carefully the instructions or to understand properly the task, you may close this tab right now and save yourself 5 minutes of your life. Please fill in the form after reading properly the instructions and making sure you got everything right. If you have any questions or difficulties please contact us.

1. First you will be given the meaning (or reference) intended to be delivered by each sentence. It may consist of one or several pieces of information

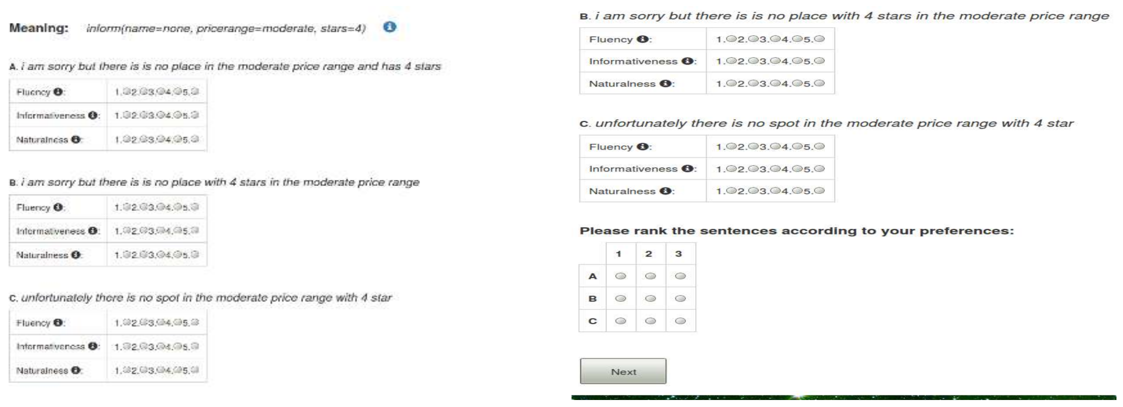
Meaning: negate(type='restaurant', food='italian', drinks='beer')

which in this case means "There is no restaurant serving Italian food and beer".

Fig. 5.1 Welcome page of the web-based evaluation interface

We provide the users with a detailed explanation of each criterion, accompanied by a number of examples. We also provide information about the semantic formalism used by the systems. Thus at each turn (new dialog act) users are presented first with a given semantic content and then with three variants of output phrases generated by three NLG systems in random order. After each phrase, users are given a 1-5 scale for each of the three criteria: fluency, informativeness and naturalness. At the end, the evaluators are asked to rank the phrases according to their preferences, before switching to the next turn/dialog act. After the 10th turn users were presented with an option to make another round of 10 phrases or close the evaluation application.

(a) Criteria scoring (b) Ranking page



Meaning: *inform(name=none, pricerange=moderate, stars=4)*

A. *i am sorry but there is is no place in the moderate price range and has 4 stars*

Fluency:	1.○2.○3.○4.○5.○
Informativeness:	1.○2.○3.○4.○5.○
Naturalness:	1.○2.○3.○4.○5.○

B. *i am sorry but there is is no place with 4 stars in the moderate price range*

Fluency:	1.○2.○3.○4.○5.○
Informativeness:	1.○2.○3.○4.○5.○
Naturalness:	1.○2.○3.○4.○5.○

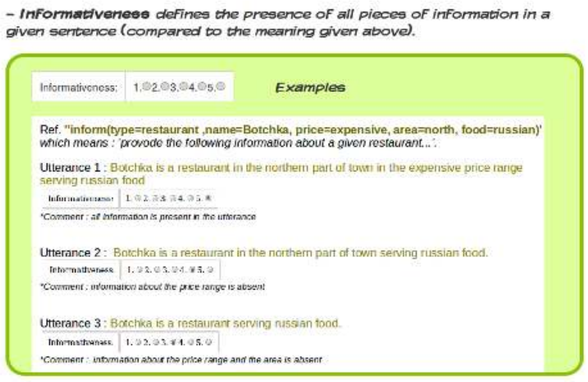
C. *unfortunately there is no spot in the moderate price range with 4 star*

Fluency:	1.○2.○3.○4.○5.○
Informativeness:	1.○2.○3.○4.○5.○
Naturalness:	1.○2.○3.○4.○5.○

Please rank the sentences according to your preferences:

	1	2	3
A	○	○	○
B	○	○	○
C	○	○	○

Fig. 5.2 Web-based evaluation interface



– ***Informativeness*** *defines the presence of all pieces of information in a given sentence (compared to the meaning given above).*

Informativeness:	1.○2.○3.○4.○5.○

Examples

Ref. **"inform(type=restaurant ,name=Botchka, price=expensive, area=north, food=russian)'**
which means : 'provode the following information about a given restaurant...'.

Utterance 1 : Botchka is a restaurant in the northern part of town in the expensive price range serving russian food

Informativeness:	1.○2.○3.○4.○5.●

Comment : all information is present in the utterance

Utterance 2 : Botchka is a restaurant in the northern part of town serving russian food.

Informativeness	1.○2.○3.○4.●5.○

Comment : information about the price range is absent

Utterance 3 : Botchka is a restaurant serving russian food.

Informativeness	1.○2.○3.●4.○5.○

Comment : information about the price range and the area is absent

Fig. 5.3 Criteria explanations in the web-based evaluation interface

5.3.3 Environment

It should be noted that in our setup the participants are not limited in assessment time, nor were they confined to a particular environment, and therefore they could perform the evaluation at their own pace. This might affect their perception of the phrases and their judgments, but we cannot assess the effect of the evaluation environment on the final results, nor its scale. There have been no studies in the domain of human-machine interaction (that we are aware of) on the change in perception of the naturalness and fluency of a phrase with time or after multiple readings. There is a possibility that what the user perceives as being correct without reflecting too much during continuous interaction with the system and to which he would pay no attention in a real-time continuous dialog (like the lack of an article, or some small disfluency), would easily catch his eye in a more relaxed atmosphere of home/work environment or after several readings.

There is another important point that should be made: the generated phrases presented to users belong to the spoken language and therefore are meant to be pronounced/heard and not read from the screen. Thus the difference in perception between written and spoken phrases is another aspect that we are not able to measure, control or take into consideration when analyzing the results.

We also would like to compare the evaluation time for native and non-native speakers; thus we register each user's processing time per phrase. Unfortunately it is difficult to analyze or draw any conclusion from this information. As we do not see the users and do not restrict them in evaluation time, we cannot be certain as to what causes the delay and how they may have distributed their assessment time.

5.4 Results

In this section we present the results of the evaluation described above. First we analyse the general outcome, i.e. the assessments of all users combined. Then we split it into target groups (native/non-native speakers) and examine the results of each target group.

Specifically we are interested in system ranking and the average score for each of the three criteria: fluency, informativeness and naturalness. In the result tables '1xm baseline' system represents the combined BLM/CRF model built on the original corpus, '1xm extend' is the same model built on the extended corpus and 'templates' is the template-based generator.

5.4.1 Agreement

In order to verify the consistency in judgments that we have obtained within each user group, we calculated Kappa agreement for each of them: native, advanced and intermediate English speakers (Table 5.1). Also we are interested in the user agreement on each criterion separately: fluency (F), informativeness (I) and naturalness (N).

Table 5.1 Inter-annotator agreement (Kappa)

User group	1xm baseline (F/I/N)	1xm extended (F/I/N)	Templates (F/I/N)
Native	0.8/0.9/0.7	0.8/0.9/0.8	0.7/0.8/0.6
Advanced	0.8/1.0/0.5	0.1/0.6/0.1	0.6/0.9/0.6
Intermediate	0.5/0.8/0.3	0.3/0.9/0.5	0.4/0.6/0.2

As we can see the agreement on informativeness is rather high in most cases except for the extended system in the 'advanced' group (and even there it is rather high: 0.6).

Informativeness is rather an objective criterion. As for fluency and naturalness, the agreement varies in different systems and different user groups, which is not surprising as the criteria are highly subjective. Also a higher agreement among the native speakers might be explained by a relatively small number of judges in this group compared to the other two.

Another important issue is a very low agreement on the fluency and naturalness of the extended system in the advanced group. This indicates that the perception of the extended system output is indeed a matter of personal user preferences (and probably their familiarity with the vocabulary used by the system) and varies from one user to another.

5.4.2 Overall results: all users combined

The distribution of users in our sample is not balanced: native speakers are the smallest group (6) and advanced speakers the largest (33), with the intermediate group in the middle (17). So the overall results are tilted towards non-natives, specifically the ’advanced’ users which comprise the largest group.

Table 5.2 Overall system ranking

Model	Rank 1	Rank 2	Rank 3
1xm baseline	51%	34%	15%
1xm extend	34%	32%	34%
templates	55%	29%	16%

Table 5.3 Overall average per criterion (1-5)

Model	Fluency	Informativeness	Naturalness
1xm baseline	4.20	4.52	4.05
1xm extend	3.61	4.11	3.37
templates	4.29	4.54	4.16

Table 5.2 shows the distribution of ranks among the three systems. It should be noted that the users had the possibility to give the same rank to two or all three systems if they considered them to have comparable performance.

As we can see ’1xm baseline’ system has somewhat similar rank distribution with the template-based system, while the extended model is seen as the best system only in 34% of cases. The same trend is observed in the distribution of average scores per criterion (Table 5.3); ’1xm baseline’ and template-based model show similar results, while the extended system is behind on all parameters. As we mentioned before these results may be largely influenced by the composition of our user sample, so we take a closer look at each group in isolation in the following sections.

5.4.3 Native speakers

Native speakers comprise the smallest group in our evaluator sample. Nevertheless their assessments are crucial for validation of our extension methodology. Table 5.4

Table 5.4 System ranking: native speakers

Model	Rank 1	Rank 2	Rank 3
1xm baseline	50%	19%	31%
1xm extend	42%	35%	23%
templates	42%	31%	27%

The small size of this user group may account for the disparity in their results. But we are interested in their assessments of the extended system and its placement in comparison to the other two systems, as well as its average naturalness and fluency scores.

Table 5.5 Average per criterion (1-5): native speakers

Model	Fluency	Informativeness	Naturalness
1xm baseline	4.11	4.57	3.92
1xm extend	4.15	4.42	4.0
templates	4.19	4.38	4.0

Interestingly, according to the users' assessments, the extended system is comparable in fluency to the template-based generator (4.15 vs 4.19) and is equivalent to the later in terms of naturalness, being at the same time slightly higher in informativeness. This is a noteworthy phenomenon: possibly, the difference in vocabulary might affect the perception of the overall informational content of the phrase for a native speaker. This contrasts with what we observed in the assessments of the non-native speakers, who oftentimes dismiss extended system output as less natural and less fluent.

In general we can observe less dispersion in judgements between different systems here, than in non-native speakers' results (see Section 5.4.4); all three systems are viewed as more or less equivalent in the quality of their outputs. Yet this general assessment may result from a relatively small number of judges in this user group.

5.4.4 Non-native speakers

In this section we take a closer look at the results of the non-native speakers group, which represents the core of our user sample.

Users with advanced level of English

This is the most representative group in our evaluation sample with a total of 36 judges. In tables 5.6 and 5.7 we can see a distinct preference for template-based system outputs, although the '1xm baseline' systems rankings and scores are very close to the template-based ones. Also we notice a comparatively low naturalness score for the extended system. This may indicate that the diversity in the vocabulary makes the utterances harder to process for a non-native speaker and make 'extended' output seem less natural than the simpler template-generated phrases which use a small vocabulary and a limited set of syntactic structures.

Table 5.6 System ranking: non-native speakers (advanced)

Model	Rank 1	Rank 2	Rank 3
1xm baseline	55%	33%	12%
1xm extend	35%	31%	34%
templates	59%	27%	14%

Table 5.7 Average per criterion (1-5): non-native speakers (advanced)

Model	Fluency	Informativeness	Naturalness
1xm baseline	4.18	4.54	4.05
1xm extend	3.62	4.15	3.37
templates	4.30	4.60	4.15

At the same time the equal distribution of the extended system rankings and 35% of phrases ranked as best outputs might suggest that we may be dealing with occasional vocabulary 'strangeness' and not a systematic phenomenon.

Also we notice a correlation between fluency and naturalness scores: phrases which are not perceived as 'fluent' consequently tend to loose in naturalness as well.

Users with intermediate level of English

Globally, the results in this group are similar to the advanced users' judgments. There are some slight variations in both the rankings and the average scores, but the overall tendencies are preserved: the extended system is ranked lower than the other two which are more or less equivalent in the criteria assessment.

Table 5.8 System ranking: non-native speakers (intermediate)

Model	Rank 1	Rank 2	Rank 3
1xm base	44%	38%	18%
1xm extend	29%	36%	35%
templates	51%	31%	18%

Table 5.9 Average per criterion (1-5): non-native speakers (intermediate)

Model	Fluency	Informativeness	Naturalness
1xm base	4.29	4.54	4.14
1xm extend	3.56	4.04	3.32
templates	4.39	4.52	4.30

However there is a tangible preference for template-based system which is ranked first in 51% of cases (against a more even distribution of ranks for the two other systems).

5.5 Outcome analysis

In this section we present some issues and phenomena that we encountered during the analysis of the results, and that we found worth a closer examination and useful for a deeper understanding of the evaluation outcome.

Subjectivity of the judgments

Assessments of the same outputs made by several users in the same target group may vary significantly. For example, consider the three outputs for produced by the template-based system (a) the 1xm baseline system (b) and the extended 1xm system (c).

(a) bochka is a hotel on cascade road serving indian food prices are around 15 pounds per person
(b) bochka is a hotel on cascade road serving indian food at about 15 pounds per person
(c) bochka is a hotel on cascade road proposing indian cuisine prices are around 15 pounds per person

One user evaluates both fluency and naturalness of the utterance (c) with 5 and ranks it as the best output, while another gives 3 and 4 respectively to (c) and ranks it as the worst and (b) as the best output. This illustrates the subjective nature of the users' perception of any given utterance.

Interpretation of the evaluation criteria

Apart from the subjective nature of output perception, the evaluation criteria themselves also appeared to be subjects to different interpretations. Despite the fact that we provided a detailed description of each criterion with examples, some users added their own understanding and vision of each criterion while performing the assessment, which is perfectly fine in itself, but should be taken into consideration while analyzing the results. We interviewed several users after the evaluation to obtain their feedback and comments on the evaluation deployment.

For example, one of the issues concerned the 'informativeness' criterion: additional information in the output which was not present in the semantics of a given dialog act was dismissed as "not relevant" by some users and consequently provoked downgrading of the informativeness score:

Semantics: $inform(name = char\ sue,\ drinks = wine,\ stars = 3)$
1xm extended: *char sue offers great wine and and it has 3 stars*

Here despite the presence of all the information in the output, the informativeness score was degraded to 4 due to the presence of *great* in *offers great wine* as, according to one user, there was no information about the quality of the wine in the semantics, so he considered this information was 'made up' by the system itself and thus not reliable, nor relevant. We cannot disagree with this remark and take it into consideration for the future improvements of our generation systems.

Another issue that we faced was the confusion with the punctuation marks:

(a) I am sorry but there is no restaurant matching your request
(b) I am sorry , but there is no restaurant that matches your request

One user reported that in cases like that punctuation was a decisive point (for him) in assigning a score to a given utterance. In this particular case the absence of a comma in the first case resulted in degrading the fluency score to 4, while the second utterance was given a 5. Most users however did not pay attention to punctuation. Although we specified that the users were dealing with the spoken dialog systems' output, presenting them in a written form triggered some sort of orthographic analysis on the part of judges in some cases, which in turn affected their assessment.

Difficulties with the semantic formalism

Some users reported their difficulties with interpreting the semantic content and matching it against the presented system outputs. However they stated that it did not affect their judgments on fluency and naturalness. They also expressed their suggestions for simplification of the semantic representation, notably reducing the number of dialog act types (in order to facilitate the evaluation process).

Perception of the extended system

One of the most notable results of this evaluation campaign was the non-native users' disinclination for the outputs of our extended systems. For example, comparing the outputs of the extended (b) and template-based (a) systems:

(a) art house hotel is a restaurant on art square serving english food near cinema
(b) art house hotel is an eating place on art square serving english dishes not far from movie house

we often encountered cases of a sharp difference in scores given by the same user, like the following (fluency/informativeness/naturalness):

(b) 5/5/5; Rank 1
(a) 2/3/2; Rank 3

We noticed that in many cases the overall output perception (by a particular user) affects all criteria at once: if the output does not seem natural, it is almost automatically scored as less fluent and less informative.

The above discussion leads us to the conclusion that, the generation system which targets a wide range of users, including non-native speakers, should ideally produce outputs which use a standard, limited vocabulary and simple syntactic structures in order to make the system-user interaction as simple and straightforward for the later as possible.

5.6 Discussion

Although automatic evaluation metrics, like METEOR or BLEU, are widely used for assessment of natural language production systems and provide useful information about the performance, they do have their limits. Human evaluation, especially the one performed by the target user group, remains the most valuable source of information about the systems' output quality. Besides providing a deeper insight into the user perspective and perception of the systems output, it may also reveal problems which are not obvious for system developers and domain experts.

We were confronted with a number of issues which made the results somewhat unexpected and difficult to analyze. It also allowed us to see the other side of our approach, notably for corpus extension, and its limits. We discuss these points in more detail in the respective subsections below.

Subjectivity of judgments

As we have repeated throughout this chapter, the evaluation performed by human judges is highly subjective; not only different users have different backgrounds and thus different perceptions of generation outputs, but also the judgments made by the same user in different conditions and evaluation environments may differ.

We did our best to ensure a sufficient variety and representativeness in our user sample to make the evaluation the least biased and to target a broader user selection. At the same time we are not able to take into account all specific user characteristics (like gender, age, etc.) while analyzing the results, but we do take into consideration the country of origin and the level of English of each evaluator.

Complexity of the task

Another important issue was the complexity of our task for untrained non-expert evaluators. One specific problem was the clarification of the assessment criteria. Our goal was to find a compromise between the granularity of explanation and the simplicity of the presentation. Yet the fine distinctions between the criteria were somewhat hard to draw in certain cases.

Moreover each user had his own subjective perception of the evaluation criteria. Another problem was, that not all judges found the semantic formalism that we used sufficiently intuitive. This might have caused problems while assessing the informativeness of the output (as we discussed in the previous section). As most of these cases did not concern the overwhelming majority of the users, but were rather occasional isolated issues, we decided to disregard them while performing the analysis of the results.

Differences in judgments: native vs non-native speakers

Different user backgrounds shaped differences in judgments. This is especially true for native and non-native English speakers and their assessment of the 'extended' system outputs: non-native speakers found it hard to deal with phrases which contained unknown words or expressions and often dismissed them as ’unnatural’, while native speakers ranked such phrases as equal or sometimes even higher than the ones from the original corpus.

This raises a question: should a dialog systems be oriented towards a particular user group or should it be generic enough to target as many users as possible? Obviously it depends on the task at hand, the context and the general purpose of the system. International dialog systems, targeting non-native speakers, might employ basic simple phrases which use standard classical English vocabulary, with no slang or local variations (like ’movie house’ in the example above).

Native speakers do appreciate diversity and variability; yet they may come from different countries, use different vocabulary and, consequently, find some words or expressions - not typical of their dialect - unusual. At the same time systems which require precision of the delivered information (like ticket reservation agents) might profit from repetitious uniform phrases; it is important to make sure that the user hears each statement multiple times in the same form and does not get confused with diverse output changing at each turn. In a less task specific or general purpose conversational agents, the extended corpus may be more appropriate for training the NLG module.

Performance of our NLG models compared to the template-based generator

The evaluation results showed that the performance of our combined BLM/CRF system built on the original corpus is comparable to that of the template-based generator according to all three criteria: fluency, informativeness and naturalness. This clearly demonstrates the potential of our data-driven approach in limited task-oriented domains. As for the extended system, the assessments of users differed in two target groups, with lower scores given by non-native speakers and comparable scores obtained from native speakers. As the later group is smaller than the former, global averages of the extended system were below those of other systems on all three criteria.

Besides the global assessment of the systems’ performance, the intrinsic evaluation campaign provided us with a useful feedback for future improvements and adjustments, notably in the area of corpus extension. We believe that an additional extrinsic evaluation could provide an even deeper insight into the users’ experience with the system in a real-time interaction environment. We leave this for future work.

Chapter 6

Conclusions and future work

In this final chapter we take a look back at the study that we carried out. In section 6.1 we discuss the results that we obtained as well as the problems that we encountered during our work. We also discuss the improvements and developments that we are planning to bring about in order to optimize and extend our final systems. Finally we outline some future perspectives and directions to follow in section 6.2.

6.1 Conclusions

In this section we look at what we achieved and learned in the course of our work. In the present thesis we address two main issues: building a robust data-driven NLG system which would replace previously used template-based generator, and diversifying its output by extending the training corpus with automatically obtained synonyms and paraphrases. As some aspects of our systems such as diversity and naturalness of the output cannot be measured automatically at development time, we perform a human evaluation involving 56 judges and get their assessments of the systems' performance. We discuss these points in corresponding subsections below.

NLG modeling

In the present work we proposed several novel approaches to natural language generation based on statistical learning.

First we investigated the idea of using n-gram translation framework for language generation. N-gram-based generation models, or bilingual language models (BLM), showed an excellent performance on our data, outperforming to a considerable degree the phrase-based baseline.

We also explored the potential of discriminative models for NLG, specifically the ones based on Conditional Random Fields. These models allow for more flexibility in their output than n-gram models, at the same time demonstrating a similar level of performance. This flexibility proved to be a complementary asset in the combination of the two models.

Finally, we built a generation pipeline which comprises n-gram and CRF models and which uses an efficient decoding framework (FST). By combining these models we obtain an additional small improvement in overall performance.

Moreover, our experiments showed that our best-performing combinations of BLM/CRF perform at the level of manually built template-based generator, which was further confirmed by judges during the evaluation campaign. Thus, one2many NLG models, which operate on single concepts and their corresponding lexical realizations, and not their predefined combinations, like it is the case in template-based systems, proved to be more robust in dealing with unseen inputs (combinations of concepts).

Corpus extension

In the second part of this thesis we explored the possibilities of automatic corpus extension, which traditionally involved manual paraphrasing, either performed by human experts or obtained via crowd-sourcing.

We implemented and tested several methods that allowed us to automate the process of extending the system vocabulary and augmenting the number of syntactic constructions available to the system. Specifically we produced new training instances by, first, replacing continuous class words in the target sentences with automatically extracted synonyms, and secondly, by extracting and integrating paraphrases for contiguous subphrases.

Having performed such two-fold extension, we obtained a larger and more diverse corpus that can be used to train not only generation models, but also any other system which uses spoken data (like language understanding).The extension paradigm itself is language-independent and can be applied to augment the size of nearly any dialogue training corpus.

Evaluation

In order to get a more extensive assessment of our generation systems and a deeper insight into the effect of the corpus extension, we performed a human evaluation, where we presented the users with the output phrases, generated by our NLG models, and asked them to rate these phrases in terms of naturalness, fluency and informativeness.

The evaluation showed that our baseline BLM/CRF system achieves the performance similar to that of the rule-based generator, even occasionally outperforming it (from the user perspective) in terms of naturalness and fluency. The evaluation also revealed a discord in the preferences of various user groups, with an overall predilection of native speakers for the extended system outputs and with the overwhelming majority of non-native speakers selecting template-based and baseline BLM/CRF model outputs as their preferred ones.

The evaluation results also indicated some new directions to follow in order to make the interactive systems more user-friendly and efficient, e.g. adjusting the system vocabulary and adding more elaborate user-adaptation functionalities.

6.2 Future work

In this section we outline some further ameliorations that we intend to implement in our current systems, as well some more general future directions for work.

NLG modeling

Concerning the core generation models there are several interesting research directions to follow. Lexicalized concepts is a promising data representation that we could not fully explore in the present work. Despite a somewhat worse performance on our data, this format certainly offers more flexibility and variability on the lexical level within the couples of concepts and their lexical realizations.

Another aspect of the core model with a potential of significant impact on the performance is the reordering, both inner-concept (in case of lexicalized corpus format) and large-scale inter-concept movement. Data-driven reordering models learned from the training corpus offer a wide range of possibilities, which are worth a deeper exploration, for example learning the reordering rules from a lemmatized or stemmed corpus or from a corpus annotated with additional factors. Lastly, our framework allows for combination of several models in a form of weighed feature functions; it makes sense to try and integrate other models which can be represented in a similar format, thus providing additional information to the decoder.

More refined corpus extension pipeline

We showed that automatic corpus extension is indeed a reliable and promising alternative to manual paraphrasing. It can be further elaborated to include, for example, large scale non-contiguous paraphrases. Another important issue is the automatic paraphrase validation which is performed by system developers in our current setup. At present our automatic filtering procedure does not reach the precision of manual validation in case of paraphrase construction. Finally, the process of template recombination that we touched upon in chapter

3 and that remained a theoretical speculation in our current work, would allow to drastically reduce the time spent on corpus building by cutting down the number of initial manually built templates.

More extensive human evaluation

To evaluate the performance of our NLG systems we carried out an intrinsic human evaluation where we asked the users to assess the output phrases in isolation, out of the general conversation context. Extrinsic evaluation, where the users would communicate with the system in a continuous dialog, should provide more information about the real-time performance of the NLG modules within a full functioning dialog system, compared to the currently used in-house template-based generator, and should help us validate our extension methodology. Such aspects of the extended system outputs as variability and diversity can be fully apprehended only in the course of a continuous dialog.

Other languages and corpora

The preliminary evaluation results (BLEU) of our NLG systems trained on the French corpus turned out to be somewhat below the results of the English-language systems. This is more likely to be the result of the inherent language differences and the degree of lexical and syntactic variability, than the default of universality in our modeling approach. Nevertheless testing our methodology on other corpora remains a prospective project.

References

Online linguistic dictionary: www.lingualinks.com, 2003.

Daniel Andrade, Masaaki Tsuchida, Takashi Onishi, and Kai Ishikawa. Synonym acquisition using bilingual comparable corpora. In *Proceedings of the International Joint Conference on Natural Language Processing (IJCNLP'13), Nagoya, Japan*, 2013.

Gabor Angeli, Percy Liang, and Dan Klein. A simple domain-independent probabilistic approach to generation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 502–512. Association for Computational Linguistics, 2010.

Abhishek Arun and Philipp Koehn. Online learning methods for discriminative training of phrase based statistical machine translation. In *Proceedings of MT Summit XI*, volume 2, page 29, 2007.

Pranjal Awasthi, Delip Rao, and Balaraman Ravindran. Part of speech tagging and chunking with hmm and crf. In *Proceedings of NLP Association of India (NLPAI) Machine Learning Contest*, 2006.

Rafael E Banchs and Marta R Costa-jussà. A semantic feature for statistical machine translation. In *Proceedings of the fifth workshop on syntax, semantics and structure in statistical translation (SSST)*, pages 126–134. Association for Computational Linguistics, 2011.

Satanjeev Banerjee and Alon Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the ACL workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005.

Srinivas Bangalore and Owen Rambow. Exploiting a probabilistic hierarchical model for generation. In *Proceedings of the 18th conference on Computational linguistics*, pages 42–48. Association for Computational Linguistics, 2000.

Michele Banko, Vibhu O Mittal, and Michael J Witbrock. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 318–325. Association for Computational Linguistics, 2000.

Colin Bannard and Chris Callison-Burch. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 597–604. Association for Computational Linguistics, 2005.

Alberto Barrón-Cedeño, Marta Vila, M Antònia Martí, and Paolo Rosso. Plagiarism meets paraphrasing: Insights for the next generation in automatic plagiarism detection. *Computational Linguistics*, 39(4):917–947, 2013.

Regina Barzilay and Lillian Lee. Learning to paraphrase: an unsupervised approach using multiple-sequence alignment. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 16–23. Association for Computational Linguistics, 2003.

Regina Barzilay and Kathleen R McKeown. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 50–57. Association for Computational Linguistics, 2001.

Regina Barzilay, Kathleen R McKeown, and Michael Elhadad. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 550–557. Association for Computational Linguistics, 1999.

John A Bateman, Robert T Kasper, Johanna D Moore, and Richard A Whitney. A general organization of knowledge for natural language processing: The penman upper model. Technical report, USC Information Sciences Institute, 1990.

Leonard E Baum and Ted Petrie. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, pages 1554–1563, 1966.

Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, pages 164–171, 1970.

Anja Belz. Automatic generation of weather forecast texts using comprehensive probabilistic generation-space models. *Natural Language Engineering*, 14(04):431–455, 2008.

Anja Belz and Ehud Reiter. Comparing automatic and human evaluation of nlg systems. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2006.

Christina Bennett and Alexander I Rudnicky. The carnegie mellon communicator corpus. In *Proceedings of the International Conference on Spoken Language Processing (ICASLP)*, 2002.

Adam L Berger, Vincent J Della Pietra, and Stephen A Della Pietra. A maximum entropy approach to natural language processing. *Computational linguistics*, 22(1):39–71, 1996.

Leonard Bloomfield. *Language*. Henry Holt, New York, 1933.

Paul Boersma and David Weenink. The towninfo spoken language understanding corpus. Edinburgh University, School of Informatics, unpublished, 2008.

Dwight Bolinger. Entailment and the meaning of structures. *Glossa*, 2(2):119–127, 1968.

Danushka Bollegala, Yutaka Matsuo, and Mitsuru Ishizuka. Measuring semantic similarity between words using web search engines. In *Proceedings of the International World Wide Web Conference (IW3C2)*, pages 757–766, 2007.

Johan Boye, Mats Wirén, and Joakim Gustafson. Contextual reasoning in multimodal dialogue systems: Two case studies. In *Proceedings of the 8th international workshop on formal semantics and pragmatics of dialogue, catalog*, pages 4–11, 2004.

Thorsten Brants and Alex Franz. Web 1t 5-gram version 1. 2006. Linguistic Data Consortium, Philadelphia.

Eric Brill. A simple rule-based part of speech tagger. In *Proceedings of the workshop on Speech and Natural Language*, pages 112–116. Association for Computational Linguistics, 1992.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85, 1990.

Stephan Busemann, Thierry Declerck, Abdel Kader Diagne, Luca Dini, Judith Klein, and Sven Schmeier. Natural language dialogue service for appointment scheduling agents. In *Proceedings of the fifth conference on Applied natural language processing (ANLP)*, pages 25–32. Association for Computational Linguistics, 1997.

Marine Carpuat. Toward using morphology in french-english phrase-based smt. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 150–154. Association for Computational Linguistics, 2009.

Alexandru Ceausu. Rich morpho-syntactic descriptors for factored machine translation with highly inflected languages as target. In *Proceedings of the Workshop on Machine Translation and Morphologically-rich langauges, University of Haifa*, 2011.

Daniel Cer, Daniel Jurafsky, and Christopher D Manning. Regularization and search for minimum error rate training. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 26–34. Association for Computational Linguistics, 2008.

Yee Seng Chan and Hwee Tou Ng. Maxsim: A maximum similarity metric for machine translation evaluation. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 55–62, 2008.

Senthilkumar Chandramohan, Matthieu Geist, Fabrice Lefèvre, and Olivier Pietquin. User Simulation in Dialogue Systems using Inverse Reinforcement Learning. In *Proceedings of the INTERSPEECH*, 2011.

Songsak Channarukul, SUSAN W McROY, and Syed S Ali. Yag: A template-based text realization system for dialog. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 9(06):649–659, 2001.

Mao Chen, Ehsan Foroughi, Fredrik Heintz, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Itsuki Noda, Oliver Obst, Patrick Riley, Timo Steffens, et al. Users manual: Robocup soccer server manual for soccer server version 7.07 and later, 2003.

Colin Cherry and George Foster. Batch tuning strategies for statistical machine translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436. Association for Computational Linguistics, 2012.

David Chiang. A hierarchical phrase-based model for statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 263–270. Association for Computational Linguistics, 2005.

David Chiang. Hope and fear for discriminative training of statistical translation models. *The Journal of Machine Learning Research*, 13(1):1159–1187, 2012.

Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and psychological measurement*, 20(1):37–46, 1960.

Jacob Cohen. Weighted kappa: Nominal scale agreement provision for scaled disagreement or partial credit. *Psychological bulletin*, 70(4):213, 1968.

Michael Collins, Philipp Koehn, and Ivona Kučerová. Clause restructuring for statistical machine translation. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, pages 531–540. Association for Computational Linguistics, 2005.

Simon Corston-Oliver, Michael Gamon, Eric Ringger, and Robert Moore. An overview of amalgam: A machine-learned generation module. In *Proceedings of the International Natural Language Generation Conference*, pages 33–40, 2002.

Josep Crego, François Yvon, and José Mariño. Ncode: an open source bilingual n-gram smt toolkit. *The Prague Bulletin of Mathematical Linguistics*, 96:49–58, 2011.

Josep M Crego and José B Marino. Extending marie: an n-gram-based smt decoder. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 213–216. Association for Computational Linguistics, 2007.

Josep Maria Crego and José B Mariño. Improving statistical mt by coupling reordering and decoding. *Machine Translation*, 20(3):199–215, 2006.

David Alan Cruse. *Lexical semantics. Cambridge textbooks in linguistics.* Cambridge, 1986.

Tri Dao, Sam Keller, and Alborz Bejnood. Alternate equivalent substitutes: Recognition of synonyms using word vectors. 2013.

Dipanjan Das and André FT Martins. A survey on automatic text summarization. *Literature Survey for the Language and Statistics II course at CMU*, 4:192–195, 2007.

Renato De Mori, Frédéric Bechet, Dilek Hakkani-Tur, Michael McTear, Giuseppe Riccardi, and Gokhan Tur. Spoken language understanding. *Signal Processing Magazine, IEEE*, 25 (3):50–58, 2008.

David DeVault, David Traum, and Ron Artstein. Practical grammar-based nlg from examples. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 77–85. Association for Computational Linguistics, 2008.

Georgiana Dinu and Marco Baroni. How to make words with vectors: Phrase generation in distributional semantics. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 624–633, 2014.

George Doddington. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc., 2002.

Zhendong Dong and Qiang Dong. Development stragegy of hownet. In *Proceedings of the Fudan Seminor on Chinese Language*, 2004.

Daniel Duma and Ewan Klein. Generating natural language from linked data: Unsupervised template extraction. *Proceedings of the Association for Computational Linguistics (ACL), Potsdam, Germany*, pages 83–94, 2013.

Michael Elhadad, Jacques Robin, and Kathleen McKeown. Floating constraints in lexical choice. *Computational Linguistics*, 23(2):195–239, 1997.

Christiane Fellbaum. *WordNet*. Wiley Online Library, 1998.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

Jeffrey Flanigan, Chris Dyer, and Jaime G Carbonell. Large-scale discriminative training for statistical machine translation using held-out line search. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (HLT-NAACL)*, pages 248–258, 2013.

Joseph L Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

Mary Ellen Foster. Automated metrics that agree with human judgements on generated output for an embodied conversational agent. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 95–103. Association for Computational Linguistics, 2008.

W Nelson Francis. A standard corpus of edited present-day american english. *College English*, pages 267–273, 1965.

Reva Freedman. Atlas: A plan manager for mixed-initiative, multimodal dialogue. In *Proceedings of the National Conference on Artificial Intelligence(AAAI-99) Workshop on Mixed-Initiative Intelligence*, pages 1–8, 1999.

Dayne Freitag, Matthias Blume, John Byrnes, Edmond Chow, Sadik Kapadia, Richard Rohwer, and Zhiqiang Wang. New experiments in distributional representations of synonymy. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 25–32. Association for Computational Linguistics, 2005.

Dimitra Gkatzia, Helen Hastie, and Oliver Lemon. Multi-adaptive natural language generation using principal component regression. *Proceedings of the International Natural Language Generation conference (INLG)*, page 138, 2014.

Arthur C Graesser, Patrick Chipman, Brian C Haynes, and Andrew Olney. Autotutor: An intelligent tutoring system with mixed-initiative dialogue. *IEEE Transactions on Education*, 48(4):612–618, 2005.

Masato Hagiwara. A supervised learning approach to automatic synonym identification based on distributional features. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Student Research Workshop*, pages 1–6. Association for Computational Linguistics, 2008.

Stefan Hahn, Patrick Lehnen, Christian Raymond, and Hermann Ney. A comparison of various methods for concept tagging for spoken language understanding. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2008.

Stefan Hahn, Marco Dinarelli, Christian Raymond, Fabrice Lefevre, Patrick Lehnen, Renato de Mori, Alessandro Moschitti, Hermann Ney, and Giuseppe Riccardi. Comparing stochastic approaches to spoken language understanding in multiple languages. *IEEE Transactions on Audio, Speech & Language Processing*, 19(6):1569–1583, 2011.

Aaron Han and Lidia Chao. Lepor: a robust evaluation metric for machine translation with augmented factors. In *Proceedings of the 24th International Conference on Computational Linguistics*, page 441, 2012.

Yulan He and Steve Young. Hidden vector state model for hierarchical semantic parsing. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, (ICASSP'03)*. IEEE, 2003.

Zhiqing Shao Hu Fanghuai and Tong Ruan. Self-supervised synonym extraction from the web. *Journal of Information Science and Engineering*, 2012.

Kuo-Chuan Huang, James Geller, Michael Halper, Yehoshua Perl, and Junchuan Xu. Using wordnet synonym substitution to enhance umls source integration. *Artificial intelligence in medicine*, 46(2):97–109, 2009.

Srinivasan Janarthanam and Oliver Lemon. Adaptive referring expression generation in spoken dialogue systems: Evaluation with real users. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 124–131. Association for Computational Linguistics, 2010.

Rohit J Kate, Yuk Wah Wong, and Raymond J Mooney. Learning to transform natural to formal languages. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 1062. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

Boris Katz, Jimmy J Lin, Daniel Loreto, Wesley Hildebrandt, Matthew W Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. Integrating web-based and corpus-based techniques for question answering. In *Proceedings of the Text Retrieval Conference (TREC)*, pages 426–435, 2003.

Stefan Kaufmann, Cleo Condoravdi, and Valentina Harizanov. Formal approaches to modality. *The expression of modality*, pages 71–106, 2006.

Soo-Min Kim and Eduard Hovy. Identifying and analyzing judgment opinions. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 200–207. Association for Computational Linguistics, 2006.

Reinhard Kneser and Hermann Ney. Improved backing-off for m-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE, 1995.

Philipp Koehn and Hieu Hoang. Factored translation models. In *Proceedings of the Conference on Empirical Methods on Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 868–876, 2007.

Philipp Koehn and Josh Schroeder. Experiments in domain adaptation for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227. Association for Computational Linguistics, 2007.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 48–54. Association for Computational Linguistics, 2003.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Philipp Koehn et al. Europarl: A multilingual corpus for evaluation of machine translation. 2002.

Ioannis Konstas and Mirella Lapata. Concept-to-text generation via discriminative reranking. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 369–378. Association for Computational Linguistics, 2012.

Angelika Kratzer. The notional category of modality. *Words, worlds, and contexts*, pages 38–74, 1981.

Klaus Krippendorff. *Content Analysis: an Introduction to its Methodology*. Beverly Hills: Sage Publications, 1980.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, 2001.

Irene Langkilde and Kevin Knight. Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 704–710. Association for Computational Linguistics, 1998.

Irene Langkilde-Geary. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proceedings of the 12th International Natural Language Generation Workshop*, pages 17–24, 2002.

Brian Langner, Stephan Vogel, and Alan W Black. Evaluating a dialog language generation system: comparing the mountain system to other nlg approaches. In *Proceedings of the INTERSPEECH Conference*, pages 1109–1112, 2010.

Thomas Lavergne, Josep Maria Crego, Alexandre Allauzen, and François Yvon. From n-gram-based to crf-based translation models. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 542–553. Association for Computational Linguistics, 2011.

Alon Lavie and Michael J Denkowski. The meteor metric for automatic evaluation of machine translation. *Machine translation*, 23(2-3):105–115, 2009.

Benoit Lavoie and Owen Rambow. A fast and portable realizer for text generation systems. In *Proceedings of the fifth conference on Applied natural language processing*, pages 265–268. Association for Computational Linguistics, 1997.

Chul Min Lee, Shrikanth S Narayanan, and Roberto Pieraccini. Classifying emotions in human-machine spoken dialogs. In *Proceedings of the International Conference on Multimedia and Expo (ICME'02)*, volume 1, pages 737–740. IEEE, 2002.

Fabrice Lefèvre. Dynamic bayesian networks and discriminative classifiers for multi-stage semantic interpretation. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2007.

Oliver Lemon. Adaptive natural language generation in dialogue using reinforcement learning. *Proceedings of the Workshop on the Semantics and Pragmatics of Dialogue.(SEM-dial)*, pages 141–148, 2008.

Douglas B Lenat and Ramanathan V Guha. *Building large knowledge-based systems; representation and inference in the Cyc project*. Addison-Wesley Longman Publishing Co., Inc., 1989.

Anton Leuski and David R Traum. Npceditor: A tool for building question-answering characters. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2010.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proceedings of the ACL-04 Workshop Text Summarization Branches Out*, pages 74–81, 2004.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. Identifying synonyms among distributionally similar words. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 3, pages 1492–1493, 2003.

Wei Lu, Hwee Tou Ng, and Wee Sun Lee. Natural language generation with tree conditional random fields. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 400–409. Association for Computational Linguistics, 2009.

Andrew L Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150. Association for Computational Linguistics, 2011.

Wolfgang Macherey, Franz Josef Och, Ignacio Thayer, and Jakob Uszkoreit. Lattice-based minimum error rate training for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 725–734. Association for Computational Linguistics, 2008.

Saad Mahamood and Ehud Reiter. Generating affective natural language for parents of neonatal infants. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 12–21. Association for Computational Linguistics, 2011.

François Mairesse and Steve Young. Stochastic language generation in dialogue using factored language models. *Computer Linguistics*, 2014.

François Mairesse, Milica Gašić, Filip Jurčíček, Simon Keizer, Blaise Thomson, Kai Yu, and Steve Young. Phrase-based statistical language generation using graphical models and active learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1552–1561. Association for Computational Linguistics, 2010.

José B Marino, Rafael E Banchs, Josep M Crego, Adria de Gispert, Patrik Lambert, José AR Fonollosa, and Marta R Costa-Jussà. N-gram-based machine translation. *Computational Linguistics*, 32(4):527–549, 2006.

Yuval Marton, Chris Callison-Burch, and Philip Resnik. Improved statistical machine translation using monolingually-derived paraphrases. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 381–390. Association for Computational Linguistics, 2009.

Andrew McCallum and Wei Li. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (HLT-NAACL)*, pages 188–191. Association for Computational Linguistics, 2003.

David D McDonald. Type-driven suppression of redundancy in the generation of inference-rich reports. In *Aspects of automated natural language generation*, pages 73–88. Springer, 1992.

Kathleen McKeown, Judith Klavans, Vasileios Hatzivassiloglou, Regina Barzilay, and Eleazar Eskin. Towards multidocument summarization by reformulation: Progress and prospects. In *Proceedings of the National Conference on Artificial Intelligence (AAAI/IAAI)*, pages 453–460, 1999.

Dennis N Mehay and Michael White. Shallow and deep paraphrasing for improved machine translation parameter optimization. In *Proceedings of the AMTA 2012 Workshop on Monolingual Machine Translation (MONOMT)*, 2012.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (HLT-NAACL)*, pages 746–751, 2013.

Bonan Min, Shuming Shi, Ralph Grishman, and Chin-Yew Lin. Towards large-scale unsupervised relation extraction from the web. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(3):1–23, 2012.

Margaret Mitchell, WA Redmond, Dan Bohus, and Ece Kamar. Crowdsourcing language generation templates for dialogue systems. *Proceedings of the International Conference on Natural Language Generation and the Special Interest Group on Discourse and Dialogue*, page 16, 2014.

Robert C Moore and Chris Quirk. Random restarts in minimum error rate training for statistical machine translation. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 585–592. Association for Computational Linguistics, 2008.

Masaki Murata, Toshiyuki Kanamaru, and Hitoshi Isahara. Automatic synonym acquisition based on matching of definition sentences in multiple dictionaries. In *Computational Linguistics and Intelligent Text Processing*, pages 293–304. Springer, 2005.

Rebecca Nesson, Alexander Rush, and Stuart Shieber. Induction of probabilistic synchronous tree-insertion grammars for machine translation. In *Proceedings of the Association for Machine Translation in the Americas*, 2006.

Eugene A Nida. Analysis of meaning and dictionary making. *International Journal of American Linguistics*, pages 279–292, 1958.

Sonja Nießen, Franz Josef Och, Gregor Leusch, Hermann Ney, et al. An evaluation tool for machine translation: Fast evaluation for mt research. In *LREC*, 2000.

Franz Josef Och. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 160–167. Association for Computational Linguistics, 2003.

Franz Josef Och and Hermann Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 295–302. Association for Computational Linguistics, 2002.

Franz Josef Och and Hermann Ney. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, 2003.

Alice H Oh and Alexander I Rudnicky. Stochastic language generation for spoken dialogue systems. In *Proceedings of the 2000 ANLP/NAACL Workshop on Conversational systems-Volume 3*, pages 27–32. Association for Computational Linguistics, 2000.

Tim Paek. Reinforcement learning for spoken dialogue systems: Comparing strengths and weaknesses for practical deployment. In *Proceedings of the Dialog-on-Dialog Workshop, INTERSPEECH*, 2006.

Santanu Pal, Pintu Lohar, and Sudip Kumar Naskar. Role of paraphrases in pb-smt. In *Computational Linguistics and Intelligent Text Processing*, pages 242–253. 2014.

Frank Robert Palmer. *Mood and modality*. Cambridge University Press, 2001.

Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135, 2008.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword fifth edition, june. *Linguistic Data Consortium, LDC2011T07*, 2011.

Rebecca J Passonneau, Susan L Epstein, Tiziana Ligorio, and Joshua Gordon. Embedded wizardry. In *Proceedings of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 248–258. Association for Computational Linguistics, 2011.

Karl Pearson. Note on regression and inheritance in the case of two parents. *Proceedings of the Royal Society of London*, 58(347-352):240–242, 1895.

Paolo Petta, Catherine Pelachaud, and Roddy Cowie. Emotion-oriented systems. *The Humaine Handbook, ISBN*, 2011.

Rosalind W Picard and Jonathan Klein. Computers that recognise and respond to user emotion: theoretical and practical implications. *Interacting with computers*, 14(2):141–169, 2002.

Christian Raymond and Giuseppe Riccardi. Generative and discriminative algorithms for spoken language understanding. In *Proceedings of the INTERSPEECH Conference*, pages 1605–1608, 2007.

Samuel Reese, Gemma Boleda, Montse Cuadros, Lluís Padró, and German Rigau. Wikicorpus: A word-sense disambiguated multilingual wikipedia corpus. 2010.

Michaela Regneri and Rui Wang. Using discourse information for paraphrase extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 916–927. Association for Computational Linguistics, 2012.

Michaela Regneri, Rui Wang, and Manfred Pinkal. Aligning predicate-argument structures for paraphrase fragment extraction.

Ehud Reiter and Robert Dale. *Building natural language generation systems*, volume 33. MIT Press.

Ehud Reiter, Somayajulu G Sripada, and Roma Robertson. Acquiring correct knowledge for natural language generation. *Journal of Artificial Intelligence Research*, pages 491–516, 2003.

Giuseppe Riccardi and Dilek Hakkani-Tür. Grounding emotions in human-machine conversational systems. In *Intelligent Technologies for Interactive Entertainment*, pages 144–154. Springer, 2005.

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of the Annual Meeting-Association For Computational Linguistics*, volume 45, page 464, 2007.

Alexander I Rudnicky, Eric H Thayer, Paul C Constantinides, Chris Tchou, R Shern, Kevin A Lenzo, Wei Xu, and Alice Oh. Creating natural dialogs in the carnegie mellon communicator system. In *Proceedings of Eurospeech*, 1999.

Maria Ruiz-Casado, Enrique Alfonseca, and Pablo Castells. Using context-window overlapping in synonym discovery and ontology extension. *Proceedings of the Conference on the Recent Advances in Natural Language Processing (RANLP-2005), Borovets, Bulgaria*, 39: 43, 2005.

Sudha Sandhya and Sinhad Chitrakala. Plagiarism detection of paraphrases in text documents with document retrieval. In *Advances in Computing and Information Technology*, pages 330–338. Springer, 2011.

Helmut Schmid. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the international conference on new methods in language processing*, volume 12, pages 44–49. Citeseer, 1994.

James Shaw. Conciseness through aggregation in text generation. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 329–331. Association for Computational Linguistics, 1995.

Zhongmin Shi, Baohua Gu, Fred Popowich, and Anoop Sarkar. Synonym-based query expansion and boosting-based re-ranking: A two-phase approach for genomic information retrieval. In *Proceedings of the Fourteenth Text REtrieval Conference (TREC 2005), NIST, Gaithersburg*, 2005.

Patrick Simianer, Stefan Riezler, and Chris Dyer. Joint feature selection in distributed stochastic learning for large-scale discriminative training in smt. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 11–21. Association for Computational Linguistics, 2012.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. A study of translation edit rate with targeted human annotation.

Matthew G Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. Ter-plus: paraphrase, semantic, and alignment enhancements to translation edit rate. *Machine Translation*, 23 (2-3):117–127, 2009.

Karen Spark Jones. *Synonymy and Semantic Classification*. Edinburgh University Press, Edinburgh, 1986.

Charles Sutton and Andrew McCallum. An introduction to conditional random fields for relational learning. *Introduction to statistical relational learning*, pages 93–128, 2006.

Ben Swanson, Elif Yamangil, and Eugene Charniak. Natural language generation with vocabulary constraints. *Proceedings of the Association for Computational Linguistics (ACL)*, page 124, 2014.

Thiago Dias Tadeu, Eder Miranda de Novais, and Ivandré Paraboni. Extracting surface realisation templates from corpora. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2010.

Ross Turner, Somayajulu Sripada, Ehud Reiter, and Ian P Davy. Generating spatio-temporal descriptions in pollen forecasts. In *Proceedings of the Eleventh Conference of the European Chapter of the Association for Computational Linguistics: Posters & Demonstrations*, pages 163–166. Association for Computational Linguistics, 2006.

Lonneke Van der Plas and Jörg Tiedemann. Finding synonyms using automatic word alignment and measures of distributional similarity. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 866–873. Association for Computational Linguistics, 2006.

Giannis Varelas, Epimenidis Voutsakis, Paraskevi Raftopoulou, Euripides GM Petrakis, and Evangelos E Milios. Semantic similarity methods in wordnet and their application to information retrieval on the web. In *Proceedings of the 7th annual ACM international workshop on Web information and data management*, pages 10–16. ACM, 2005.

Wolfgang Wahlster. *SmartKom: foundations of multimodal dialogue systems*, volume 12. Springer, 2006.

Marilyn A. Walker. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, pages 387–416, 2000.

Rui Wang and Chris Callison-Burch. Paraphrase fragment extraction from monolingual comparable corpora. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 52–60. Association for Computational Linguistics, 2011.

Tong Wang and Graeme Hirst. Extracting synonyms from dictionary definitions. In *Proceedings of the Conference of the Recent Advances in Natural Language Processing (RANLP)*, pages 471–477, 2009.

Wei Wang, Klaus Macherey, Wolfgang Macherey, Franz Och, and Peng Xu. Improved domain adaptation for statistical machine translation. In *Proceedings of the 10th biennial conference of the Association for Machine Translation in the Americas (AMTA)*, 2012.

Leo Wanner. Report generation. In *Handbook of Natural Language Processing, Second Edition*. CRC Press, Taylor and Francis Group, 2010.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *preprint arXiv:1508.01745*, 2015.

Michael White and Ted Caldwell. A practical, extensible framework for dynamic text generation. In *Proceedings of the Ninth International Workshop on Natural Language Generation (INLG)*, pages 238–247, 1998.

Billy Wong and Chunyu Kit. Atec: automatic evaluation of machine translation via word choice and word order. *Machine Translation*, 23(2-3):141–155, 2009.

Billy Tak-Ming Wong. Semantic evaluation of machine translation. In *Proceedings of the Language Resources and Evaluation Conference (LREC)*, 2010.

Yuk Wah Wong. *Learning for semantic parsing and natural language generation using statistical machine translation techniques*. ProQuest, 2007.

Yuk Wah Wong and Raymond J Mooney. Generation by inverting a semantic parser that uses statistical machine translation. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics - Human Language Technologies (HLT-NAACL)*, pages 172–179, 2007.

Dekai Wu and Pascale Fung. Semantic roles for smt: a hybrid two-pass model. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 13–16. Association for Computational Linguistics, 2009.

Hua Wu and Ming Zhou. Optimizing synonym extraction using monolingual and bilingual resources. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 72–79. Association for Computational Linguistics, 2003.

Joern Wuebker, Arne Mauser, and Hermann Ney. Training phrase translation models with leaving-one-out. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 475–484. Association for Computational Linguistics, 2010.

Fei Xia and Michael McCord. Improving a statistical mt system with automatically learned rewrite patterns. In *Proceedings of the 20th international conference on Computational Linguistics*, page 508. Association for Computational Linguistics, 2004.

Kenji Yamada and Kevin Knight. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 523–530. Association for Computational Linguistics, 2001.

Xiaohao Yang and Jia Liu. Deep belief network based crf for spoken language understanding. In *Proceedings of the 9th International Symposium onChinese Spoken Language Processing (ISCSLP)*, pages 49–53. IEEE, 2014.

Takayoshi Yoshimura, Keiichi Tokuda, Takashi Masuko, Takao Kobayashi, and Tadashi Kitamura. Simultaneous modeling of spectrum, pitch and duration in hmm-based speech synthesis. In *Proceedings of EUROSPEECH*, pages 2347–2350, 1999.

Andreas Zollmann and Ashish Venugopal. Syntax augmented machine translation via chart parsing. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 138–141. Association for Computational Linguistics, 2006.